UNIVERSITÉ DE MONTRÉAL

MATHEMATICAL MODELING AND OPTIMIZATION APPROACHES FOR SCHEDULING THE REGULAR-SEASON GAMES OF THE NATIONAL HOCKEY LEAGUE

ELIVELTON FERREIRA BUENO
DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(MATHÉMATIQUES DE L'INGÉNIEUR)
AOÛT 2014

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

MATHEMATICAL MODELING AND OPTIMIZATION APPROACHES FOR SCHEDULING THE REGULAR-SEASON GAMES OF THE NATIONAL HOCKEY LEAGUE

présentée par : BUENO Elivelton Ferreira
en vue de l'obtention du diplôme de : Philosophiæ Doctor
a été dûment acceptée par le jury d'examen constitué de :

M. ROUSSEAU Louis-Martin, Ph.D., président
M. GENDREAU Michel, Ph.D., membre et directeur de recherche
M. FERLAND Jacques, Ph.D., membre et codirecteur de recherche
Mme LAHRICHI Nadia, Ph.D., membre
M. TRICK Michael A., Ph.D., membre

DEDICATION

To Carmelucia Ferreira Bueno

ACKNOWLEDGMENTS

I would like to express my deep gratitude to Professor Michel Gendreau and Professor Jacques Ferland, my research supervisors, for their guidance, encouragement and support through all the ups and downs of this journey. My grateful thanks are also extended to my other committee members, Professor Louis-Martin Rousseau, Professor Nadia Lahrichi, and Professor Michael Trick, who were more than generous with their expertise and precious time.

I owe thanks to Professor Marco Antonio Figueiredo Menezes, from the Pontifical Catholic University of Goiás, who introduced me to Operations Research; and also to Professor Nelson Maculan, from the Federal University of Rio de Janeiro, who encouraged me to dive deeper into this striking field.

Many friends have helped me stay sane through these years of study in Montreal. Among them, I would like to offer my special thanks to Dariusz Trzaniec and to the families of Emilia Rei and José Tomé, who embraced me as their own.

I am also grateful to Leizer de Lima Pinto, who has been as a supportive brother to me, and to the lovely Príscila Valcácio, who was waiting for me when I did research and wrote.

Finally, I would like to thank the Brazilian National Council for the Improvement of Higher Education (CAPES) for their financial support during three years of this research study.

RÉSUMÉ

La Ligue nationale de hockey (LNH) est une association sportive professionnelle de hockey sur glace regroupant des équipes du Canada et des États-Unis. Chaque année, la LNH dois compter sur un calendrier de haute qualité concernant des questions économiques et d'équité pour les 1230 matchs de sa saison régulière. Dans cette thèse, nous proposons le premier modèle de programmation linéaire en nombres entiers (PLNE) pour le problème de la planification de ces matchs. Basé sur la littérature scientifique en planification des horaires sportifs, et aussi sur un raisonnement pratique, nous identifions et soulignons des exigences essentielles et des préférences qui doivent être satisfaites par des calendriers de haute qualité pour la LNH. La construction de tels calendriers, tout comme la planification des horaires sportifs en général, s'avère une tâche très difficile qui doit prendre en compte des intérêts concurrents et, dans plusieurs cas, subjectifs. En particulier, les expérimentations numériques que nous décrivons dans cette étude fournissent des évidences solides suggérant qu'une approche basée sur la PLNE est actuellement incapable de résoudre des instances de taille réaliste pour le problème. Pour surmonter cet inconvénient, nous proposons ensuite un algorithme de recherche adaptative à voisinage large (ALNS) qui intègre à la fois des nouvelles stratégies et des heuristiques spécialisées provenant de la littérature scientifique. Afin de tester cette approche, nous générons plusieurs instances du problème. Toutes les instances sont basées sur les calendriers officiels de la LNH et, en particulier, utilisent les dates de matchs à domicile de chaque équipe comme des dates de disponibilité de son aréna. Dans les situations les plus difficiles, la disponibilité des arénas est rare ou est à son minimum. Dans tous les cas, en ce qui concerne les indicateurs de qualité soulevés, l'algorithme ALNS a été capable de générer des calendriers clairement meilleur que leur correspondants adoptés par la LNH. Les résultats obtenus suggèrent que notre approche pourrait certainement permettre aux gestionnaires de la LNH de trouver des calendriers de meilleur qualité par rapport à une variété de nouvelles préférences.

ABSTRACT

The National Hockey League (NHL) is a major professional ice hockey league composed of 30 teams located throughout the United States and Canada. Every year, the NHL must rely on a high-quality schedule regarding both economic and fairness issues for the 1230 games of its regular season. In this thesis, we propose the first integer linear programming (IP) model for the problem of scheduling those games. Based both on the pertinent sports scheduling literature and on practical reasoning, we identify and point out essential requirements and preferences that should be satisfied by good NHL schedules. Finding such schedules, as many other sports scheduling problems, is a very difficult task that involves several stakeholders with many conflicting, and often subjective, interests. In fact, computational experiments that we describe in this study, provide compelling evidence that an IP approach is currently unable to solve instances of realistic size for the problem. To overcome such drawback, we propose then an Adaptive Large Neighborhood Search (ALNS) algorithm that integrates both novel strategies and specialized heuristics from the scientific literature. To test the approach, we generate instances based on past NHL schedules and on a given number of arena-available dates that are suitable for the home games of each team. In the most challenging instances, availability of arenas is scarce or at its minimum. In all cases, regarding the identified concerns, the ALNS algorithm was able to generate much better schedules than those implemented by the NHL. Results obtained suggest that our approach could certainly identify unnecessary weakness in NHL schedules, makes the NHL managers aware of better schedules with respect to different requirements, and even lead them to consider other desired features they might not have previously taken into account.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

APPENDIX

CHAPTER 1

INTRODUCTION

The National Hockey League (NHL), one of the four leading professional sports leagues in North America, includes 30 teams located throughout the vast territories of the United States and Canada. Every team plays 82 games, 41 at its own home arena and 41 away (at arenas of its opponents), over slightly more than six months of the NHL regular season, which usually starts in early October each year. By that time, a schedule for all the 1230 games must already have been determined. As for any major sports league, the NHL must rely on high-quality schedules in order to make games more attractive so as to increase revenues from sources that include broadcasting rights and game attendance.

Obviously, good schedules should also take into account some fairness issues, such as reduction of fatigue for players and minimization of travel distance for teams. Because of many conflicting, and often subjective, interests from several stakeholders, including players' association, fans, teams' owners, media, and officials' association, finding a good schedule for the games of the NHL is a very difficult task.

In fact, sports scheduling problems in general are very challenging and many optimization techniques have been used to deal with them. The recent annotated bibliography by Kendall *et al.* (2010) references over 160 papers on sports scheduling, most of them having been published in the past decade alone. Also, Knust (2014) maintains an up to date collection of (currently more than 250 references) on different topics of sports scheduling and classifies them according to different models, methods, and sports disciplines. Practical papers have addressed many sports disciplines, such as soccer, baseball, golf, cricket, basketball, and ice hockey. Overall, those works point out that both artificial and real-world instances of sports scheduling problems are often surprisingly hard to be solved, even for leagues involving only a few teams.

This thesis focuses on the task of scheduling the games of the NHL regular season. We refer to that task as the *NHL scheduling problem* (the same term used in Costa, 1995). Although the literature on practical studies of sports scheduling has widely dealt with cases where, as for the NHL, a specific venue is associated to each team in the league, they have often been limited to round-robin tournaments, which is not the general structure of the NHL regular season. In fact, rather than every team playing all others a fixed number of times, an NHL team usually plays more often against opponents located in its own geographical area. And even though the exact number of games, and the places, that teams play one another are

determined beforehand, there is no predefined structure on the course of the games. Indeed, during each week different teams may play different numbers of games, usually any number in the range from two to five, and for any day, the number of games that the teams will individually have played will in general be also different from each other. Therefore, the round structure, so widely considered in other studies, is not suitable for the NHL scheduling problem. In addition, unlike most cases in the sports scheduling literature, where the ideal pattern for any team is usually to alternate between home and away games, it is allowed (and even preferred) that an NHL team visits several opponents on the road without a return to home. Such long "road trips" would thus lead to a reduction on the overall travelling distances for the teams. Nevertheless, because the home locations of the NHL teams are unevenly scattered throughout a vast region, some teams have to travel much further than others in order to complete their individual schedules. This unfavorable situation for the most geographically isolated teams could be somewhat aggravated or relieved, depending on how games are scheduled for those teams. In practice, travel distance is then one of the issues to be addressed in the NHL scheduling problem. In general, the requirements found in practical studies of sports scheduling are related to economics and fairness. For example, Bartsch *et al.* (2006) highlight the main issues dealt in several cases, including how well-distributed the most attractive games are over the season, availability of arenas, and number of rest days between two consecutive games. Nurmi *et al.* (2010) propose a framework for highly-constrained sports scheduling problems where they enumerate 36 types of constraints found in various sports leagues. Those constraints, however, cover only part of the requirements for a good NHL schedule.

Despite the recent growing scientific interest in sports scheduling, only a few researchers have addressed the NHL scheduling problem. The first systematic study leading to a successful approach was carried out in the late 1980s by Fleurent (1987), which was then (partially) reported in the paper by Ferland and Fleurent (1991). They developed a support system made up of various specialized heuristic procedures to help an expert to schedule the games in an interactive manner. They also present the first mathematical model for the problem. Unfortunately, that model considers only certain aspects of their own description of the problem. And even though only binary variables were used, it turns out to be a non-linear model. A few years later, Costa (1995) proposed a hybrid approach that combines Tabu Search and a genetic algorithm in order to solve combinatorial optimization problems in general, and then, he used the NHL scheduling problem to illustrate the effectiveness of his approach. In both studies, the descriptions of the problem enumerate several issues regarding each team, such as availability of arenas, number of games over a few consecutive days, duration and number of away games in a row, number of days between games involving the same pair of oppo-

nents, and daily travel distances. On the whole, their strategy consists in allowing violations of only some constraints and penalizing them in the objective function of a minimization problem. However, comparing one study to the other, the constraints are not considered in the same manner. In fact, there is no general agreement on a precise statement of the problem, which is typical in practical cases of sports scheduling, since many complex goals are often involved. In particular, Ferland and Fleurent (1991) give exceptional attention to the availability of arenas, as some teams may hardly be able to provide the minimum number of arena-available dates for its home games, and their approach never schedule a team to play home on a day that has not been provided to the scheduler by that team. On the other hand, Costa (1995) allows violations for the constraints on the availability of arenas, and for example, regarding the 1993-94 regular season, when the NHL was formed by only 24 teams but every team already had to play 41 games at home, his best result shows that, even though at least 56 arena dates (thus 15 "extra" dates) had been provided by each team, more than a hundred games were surprisingly scheduled on days when the arena of the home team was not available. Finally, in a more recent work, Craig *et al.* (2009) describe and analyze a system that uses a multi-objective evolutionary algorithm to schedule the games of the NHL regular season. The system is reported to produce a set of schedules that offer a range of trade-offs across the following three objectives: minimizing the total distance that all the teams must travel; avoiding the unfair situation where only one of the opponents in a game must travel a long distance over a short period of time in order to play that game; and minimizing streaks of either more than three home games or more than three away games for a team. Their assumption that, in general, streaks longer than three should be avoided is not completely consistent with the previous studies. Indeed, at least for long distance visits, both Ferland and Fleurent (1991) and Costa (1995) engage in creating road trips made up of as many as seven consecutive away games. In our view, however, the main limitation in the study by Craig *et al.* (2009) is that they overlook essential constraints of the problem, specially the availability of arenas. In particular, a team might be scheduled to play home on any day of the regular season, and the only constraint they explicitly mention is that in a feasible schedule a team cannot be assigned to play more than once a day.

This thesis revisits the NHL scheduling problem, describing some basic requirements for good NHL schedules, and present two approaches that we are proposing to construct such schedules. One approach is based on an integer linear programming (IP) formulation that we are introducing for the NHL scheduling problem and uses a commercial state-of-the-art solver to exactly solve instances of moderate size of the problem. The other approach is an adaptive large neighborhood search (ALNS) to approximately solve instances of practical size.

In particular, this thesis reinforce the importance of crucial issues that have been highlighted in the seminal work by Fleurent (1987), as for example the scarce availability of arena dates to hold the home games of the teams. Some computational experiments on a C++ implementation that uses the solver CPLEX® are reported on several variants of the proposed IP formulation. Such variants arise from different constraints being relaxed and their violations penalized in the objective function. Because, in general, those variants can enable the solver to deal only with moderate size of instances of the problem, other computational experiments are described on the ALNS as an alternative approach to solve instances of realistic size. For this approach, another model is described, in which most of the constraints are allowed to be violated during the solution process and a penalty value is introduced into an objective function that evaluates the "cost" of a schedule as a weighted summation of the constraint violations. The ALNS tries then to generate schedules that minimize the value of the objective function. Unlike the approaches by Ferland and Fleurent (1991) and by Costa (1995), where most of the requirements are treated as "hard" constraints (which must not be violated), our strategy is then more flexible, allowing easier exploration of a wide variety of schedules during the solution process. In addition, by penalizing violations of constraints in the objective function, we can mimic different degrees of preferences with the use of different penalty weights.

This thesis also describe the creation of several instances based on official NHL schedules. The reported results are very promising as, in particular, the ALNS is able to solve to optimality all instances that have only six extra arena dates per team. It accounts that even by only rescheduling the games to the same arena dates utilized in official NHL schedules, in general, our solutions violate much fewer constraints of the model. But the results also reinforce the importance of having (even just a few) extra arena dates per team. The final remarks on this study mention our confidence that the ALNS approach is able to identify unnecessary weakness in the NHL schedules, makes the NHL managers aware that the schedules could be improved with respect to different requirements, and even lead them to consider other desired features that they might not have previously taken into account.

The remainder of this thesis is organized as follows. In Chapter 2, we present some basic terminology, and a short review of the literature about the NHL scheduling problem and also about more general sports scheduling problems. We describe the NHL scheduling problem in Chapter 3, specifying the constraints that we consider in this study. An IP formulation is then proposed for each of those constraints in Chapter 4, and computational results are reported on different models that are based on the proposed formulations. The generation of the instances we have used in the experiments are also described in Chapter 4. In Chapter 5, we propose an ALNS algorithm, which also includes a review of the heuristics introduced

by Fleurent (1987), and outline the results obtained with the ALNS for instances of realistic sizes. Finally, in Chapter 6, we draw some concluding remarks and suggest future directions for this study.

CHAPTER 2

LITERATURE REVIEW

The NHL scheduling problem considered in this thesis belongs to the broader field of *sports scheduling*, which can be traced back over more than four decades. In general, problems in sports scheduling are very challenging and they have attracted the interest of a number of researchers from different fields, especially graph theory, combinatorial optimization, and applied mathematics.

The annotated bibliography by Kendall *et al.* (2010) references over 160 papers on sports scheduling, most of them having been published in the past decade alone. For pertinent surveys, we refer the reader to Easton *et al.* (2004), which review the main issues that have been considered in the literature prior to 2004; to Drexl and Knust (2007), which survey graph-based models and introduce resource-based models; and to Rasmussen and Trick (2008), which propose an unified terminology and review the literature on round-robin tournaments.

Apart from that, Knust (2014) maintains an up to date collection of references (currently with more than 250 papers) on different topics of sports scheduling and classifies them according to different models, methods, and sports disciplines. The reported approaches include integer programming (more than 60 papers), constraint programming (25), heuristics (16), and metaheuristics (30). Practical papers have addressed several sports disciplines, such as soccer (more than 30 papers), baseball (12), basketball (12), golf (6), cricket (6), and ice hockey (6). Overall, those works point out that both artificial and real-world instances of sports scheduling problems are often surprisingly hard to be solved, even for leagues involving only a few teams.

In this chapter, we provide some basic terminology used in sports scheduling, and as a way to highlight common issues and challenges arising in several practical applications, we discuss a few problems that have been widely addressed in the related literature. We also outline typical approaches reported on other studies and point out their relevance (or lack thereof) to the scheduling of the NHL regular-season games. Although the NHL scheduling problem is only described later on, in Chapter 3, we confront here all the past known studies concerning this particular application.

On the whole, we focus on sports scheduling for league tournaments in which every team (or club) is assumed to have an associated venue where games between two opposing teams take place. Such is, of course, the case of the NHL scheduling problem, where every team is

considered to own a specific arena and each game is played at the arena of one of the two opposing teams in the game. In particular, studies in other contexts, as that of the so-called Balanced Tournament Design Problem (Lamken, 1990), in which all the teams share the facilities holding the games are out of the scope of this thesis.

2.1 Basic terminology

For any particular game, the team that owns the venue where that game takes place is said to be the *home team*, while its opponent is called the *away team*. A game is said to be a *home game* with regard to the team that owns the venue where that game is to be held, but it is called an *away game* with respect to the other team. A *road trip* (or simply *trip*) for a team is a sequence of consecutive away games which that team plays without returning home in-between. The team corresponding to a certain road trip is referred to as the *traveling team*. Similarly, a sequence of consecutive home games (with no away game in-between) for a team is called a *home stand*.

Sports scheduling problems are usually classified into two large groups: temporally constrained and temporally relaxed problems (Kendall *et al.*, 2010). Here, we refer to them as *time-constrained* and *time-relaxed* problems, respectively (as in, e.g., Knust, 2010). This classification is based on the distribution of the games among the available time slots, the so-called *rounds*, of the tournament. By definition, no team plays more than a single game during each round. In the time-constrained problems the number of rounds is at its minimum with respect to the existence of a feasible distribution of the games, which defines a *compact schedule*. Each team plays then exactly one game in each round of a compact schedule for a league with an even number of teams. On the other hand, in time-relaxed problems more rounds than that minimum threshold is available, and a schedule is said to be *relaxed* for such cases. If a team is not scheduled to play during a certain round then the team is said to have a *bye* on that round.

As noted by Easton *et al.* (2004), nearly all the literature on the design of sport schedules deals with models for round-robin tournaments, to which we draw considerable attention in this chapter. There are of course other structures, as for example the Swiss tournament, *ladder*, knockout and double elimination (Rokosz, 2000), but they are out of the scope of this thesis.

2.1.1 Round-robin tournaments

A competition where the teams confront one another a fixed number of times is called a *round-robin tournament* (RRT). In most tournaments, especially for major professional sports

leagues, each team plays two times against every other team (as noted by, e.g., Rasmussen and Trick, 2008). This is particularly referred to as a *double round-robin tournament*, but *single*, *triple*, and *quadruple* round-robin tournaments, where respectively one, three, and four games are played by each pair of teams, also occur in several leagues around the world.

As an example, all national soccer championships in Europe uses an RRT during their regular season (stage). In particular, among the 25 European soccer competitions for which Goossens and Spieksma (2011) overview schedule formats, a total of 19, 4, and 2 competitions follows a double, a triple, and a quadruple RRT, respectively.

Scheduling an RRT consists in determining the round and the venue in which every game will be played. In its most common form, where the problem is time constrained, the basic sports scheduling problem for a league can be stated as follows. Given a league consisting of an even number $n \geq 2$ of teams identified by the integers in the set $T = \{1, 2, \ldots, n\}$, and a tournament where each team must play $\ell \geq 1$ times against every other team, assign all the $n(n-1)\ell/2$ games of the tournament to its $(n-1)\ell$ rounds by determining, for each $k \in \{1, 2, \ldots, (n-1)\ell\}$ and for each $i \in T$, which opponent $j \in T \setminus \{i\}$ plays against team i in the kth round, and also determining whether i plays at home (so that j plays away at i's venue) or i plays away at j's venue (so that j plays at home) in that round. In cases where the league consists of an odd number n' of teams, a "dummy team" $n'+1$ is introduced (as to have $n'+1 = n$ in the preceding statement), and an actual team is considered to have a "bye" whenever it is assigned to play against $n'+1$.

While an RRT has the advantage of ensuring that all teams individually play the same number of games, the number of teams in the league determines the length of the respective season (in number of rounds). This might be seen as a drawback if the league consists of either only a few or relatively many teams.

In a recent study, Larson and Johansson (2014) analyze the expansion of an RRT in which adding a single game for each pair of teams as to preserve an RRT format would cause a too high increase (50%) in the total number of games in the schedule. In particular, they report that the 14 team owners in the Elitserien (the top Swedish handball league) considered its traditional 26-game double RRT to be too short, but a possible 39-game triple RRT was deemed too long. Elitserien decided then to expand its season by splitting the league into two divisions of seven teams each, and by adopting a schedule where an intra-divisional single RRT is played before starting a double RRT between all teams in the league. The first half of that double RRT is, in turn, identical to the second half, except for the location of each game, which alternates between the homes of the two corresponding teams. In the literature, this is known as a *mirrored tournament*, a structure used, for instance, by several soccer leagues in South America. A mirrored double RRT is usually addressed as two consecutive

tournaments, each one being a single RRT.

This asymmetric format, in which pairs of teams play different numbers of games between them if they are either from the same division or from two distinct divisions, occurs in many sports leagues, especially in North America. Indeed, in all the leading major professional sports leagues in the United States and Canada, which include the Major League Baseball (MLB), the National Football League (NFL), the National Basketball Association (NBA), and also the National Hockey League (NHL), the number of games for each pair of teams is determined, among other factors (like the standings from the previous season), by their respective conference and divisional alignment. Despite several motivations in favor of an asymmetric format, including the benefit of minimizing travel distances for teams and promote local rivalries (DePalma, 2004; Havard, 2014), the resulting "unbalanced schedules" might rise concerns about the fairness of the competition (Lenten, 2013).

In the case of the NHL, changes in the number of games that teams play against each other was extensively investigated by Fleurent and Ferland (1993) in response to expansions of the NHL in the early 90s. They used IP to generate possible season structures for different numbers of teams and numbers of games per team. Obviously, the focus of that study is then part of the data provided for the NHL problem of this thesis.

Nowadays, because the number of teams is currently not the same for all divisions of the NHL, the teams are especially susceptible to unfair schedules. To be more precise about the structure of the league, the NHL teams are split into four divisions of either seven or eight teams: two seven-team divisions form one conference, and two eight-team divisions form another conference. As for the structure of the season, it is characterized as follows. The teams in one conference play a double RRT against the teams in the other conference. The teams in one division, which is inside a particular conference, play a triple RRT against the teams in the other division inside the same conference. By contrast, the teams inside any given division does not play an RRT: each team plays either four or five games against the other teams in its own division. Apart from that, there is no particular sequence of rounds associated to either intradivision or interdivisional games, and thus the NHL regular season structure is different from the Elitserien structure that we mentioned earlier (Larson and Johansson, 2014). Even the classical notion of "rounds" would be inaccurate in the context of the NHL scheduling problem, as at any time before the end of the tournament the teams will normally have played different numbers of games.

Despite the recent growing scientific interest in sports scheduling, very few researches have been carried out on the NHL scheduling problem. Generally speaking, this problem consists in creating schedules for the regular-season games of the NHL, which are subject to several constraints involving unavailability of some arenas on most days, limitation on the

number of games a team can play over a few days, requirement of at least a certain number of days between two games involving the same opposing teams, and minimization of traveling distances, to mention but a few. While the NHL scheduling problem will be extensively described only in Chapter 3, we address now the literature on this particular application in sports scheduling.

2.2 Literature on the NHL scheduling problem

The first systematic study leading to a successful approach to scheduling the NHL games was carried out in the late 1980s by Fleurent (1987), which is partially reported in the paper by Ferland and Fleurent (1991). Their work was motivated by the disappointing experience of Fraser (1982), who had developed a rather inflexible generator of road trips that is reported to be unable to produce feasible schedules for all the NHL games.

In his study, Fraser (1982) had in fact concluded that *"due to the unpredictability of some constraints and the human relationships involved in others, its impractical to expect that a computer program will ever produce a final schedule which would require no tuning or adjustment"* (as cited in Costa, 1995). To overcome this drawback, Ferland and Fleurent (1991) highlight the importance of considering an approach which would allow an expert to include in the scheduling process his own experience and all relevant requirements he might have. They develop then a support system made up of various specialized heuristic procedures to help a person to schedule the games in an interactive manner.

Their heuristics (which will be described in more detail in Chapter 5) can be outlined as follows, where a "free game" is a game that has not yet been scheduled at a specific time of the scheduling process.

— *Forced-trip heuristic* Schedule free games by first identifying long periods of days with no arena-available dates for a team, and then, by assigning the respective team to visit, in a single road trip, at most seven of its distant opponents exclusively during the corresponding period of arena unavailability.

— *Forced-home heuristic* Schedule, for each road trip lasting more than one week, two home games for the respective traveling team: one game on its last (latest) arena-available date before the trip, and the other, on its first arena date after the trip.

— *Free-trip heuristic* Schedule all currently free games opposing teams based far from each other by first identifying, for each team, at most three of its distant opponents to be visited in a single trip, and then, by scheduling every one of those long-distance trips on a period that does not violate the hard constraints of the problem and that contains the minimum possible number of arena-available dates of the visiting team.

— *Weekend-game heuristic* Schedule as many free games as feasibly possible respectively on Saturdays, on Fridays, and on Sundays, by trying not to schedule a team to play away on a weekend containing some of its own arena-available dates, and by never putting a team to play an away game within a period when it has already been scheduled to visit distant opponents.

— *Weekday-game heuristic* Schedule as many free games as feasibly possible on weekdays from Monday through Thursday, by trying not to schedule a team to play away on its own arena-available dates, and by never assigning days from periods of long-distance visits for the teams involved in the game being scheduled.

— *Exchange heuristic* Schedule as many free games as feasibly possible by allowing some of the previously scheduled short-distance visits to be rescheduled on alternative days.

In addition, Ferland and Fleurent (1991) propose the first mathematical model for the NHL scheduling problem. Unfortunately, that model does not consider all aspects of their own description of the problem. For example, different requirements over road-trip games for a team are not taken into account. And even though only binary variables are used, it turns out to be a non-linear model that involves, for example, products of multiple variables.

A few years later, Costa (1995) proposed a hybrid approach that combines Tabu Search and a genetic algorithm in order to solve combinatorial optimization problems in general, and then, he used the NHL scheduling problem to illustrate the effectiveness of his approach.

In both of those research studies, the descriptions of the problem enumerate several issues regarding each team, such as availability of arenas, number of games over a few consecutive days, duration and number of away games in a row, number of days between games involving the same pair of opposing teams, and travel distances. On the whole, their strategy consists in allowing violations of some constraints but penalizing them in the objective function of a minimization problem.

However, comparing one study to the other, the constraints are not considered in the same manner. In fact, there is no general agreement on a precise statement of the problem, which is typical in practical cases of sports scheduling, since many complex goals are often involved in the process.

In particular, Ferland and Fleurent (1991) give exceptional attention to the availability of arenas, as some teams may hardly be able to provide the minimum number of arena-available dates for its home games, and their approach never schedules a team to play at home on a day that has not been provided to the scheduler by that team. On the other hand, Costa (1995) allows violations for the constraints on the availability of arenas.

As an example, regarding the 1993-94 regular season, when the NHL was formed by only 24 teams but every team already had to play 41 games at home, the best result by Costa

(1995) shows that, even though at least 56 arena dates (thus 15 "extra" dates) had been provided by each team, more than a hundred games were surprisingly scheduled on days when the arena of the home team was supposed not to be available.

Finally, in a more recent work, Craig *et al.* (2009) describe and analyze a system that uses a multi-objective evolutionary algorithm to schedule the games of the NHL regular season. The system is reported to produce a set of schedules that offer a range of trade-offs across the following three objectives: minimizing the total distance that all the teams must travel; avoiding the unfair situation where only one of the rivals in a game must travel a long distance over a short period of time in order to play that game; and minimizing streaks of either more than three home games or more than three away games for a team. In addition, as in Costa (1995), they allow a team to be scheduled to play more than one game on the same day during the solution process, but eventually discard any schedule where such infeasible setting happens.

The assumption, in Craig *et al.* (2009), that streaks longer than three should be avoided is not completely consistent with the previous studies. Indeed, at least for long distance visits for a team, both Ferland and Fleurent (1991) and Costa (1995) engage in creating road trips made up of as many as seven consecutive away games, which in turn seems more in line with actual NHL schedules. Evidence for this is in tables 2.1 and 2.2.

In our view, however, the main limitation in the study by Craig *et al.* (2009) is that they overlook essential constraints of the problem, specially the availability of arenas. A team might, for instance, be scheduled to play home on any day of the regular-season period, and the only constraint they explicitly mention is that in a feasible schedule a team cannot be

Table 2.1 *Length of home stands in official NHL schedules* Frequency of home stands by their individual number of games ($\#g$) in the official schedules of past seasons

$\#g$	2009-10		2010-11		2011-12		2012-13		2013-14	
1	214	(42.04%)	221	(45.29%)	216	(42.19%)	139	(40.29%)	236	(40.48%)
2	149	(29.27%)	147	(30.12%)	145	(28.32%)	104	(30.14%)	182	(31.22%)
3	75	(14.73%)	64	(13.11%)	88	(17.19%)	62	(17.97%)	86	(14.75%)
4	40	(7.86%)	27	(5.53%)	37	(7.23%)	22	(6.38%)	38	(6.52%)
5	21	(4.13%)	14	(2.87%)	14	(2.73%)	12	(3.48%)	31	(5.32%)
6	6	(1.18%)	10	(2.05%)	8	(1.56%)	3	(0.87%)	7	(1.20%)
7	4	(0.79%)	3	(0.61%)	2	(0.39%)	3	(0.87%)	2	(0.34%)
8			1	(0.20%)	2	(0.39%)				
9									1	(0.17%)
10			1	(0.20%)						

assigned to play more than once a day.

In chapters 3 and 5 of this thesis we explore in more details the research studies by Ferland and Fleurent (1991) and by Costa (1995), including our remarks on their models for the NHL scheduling problem and also on the solution approaches they proposed to solve it.

To provide a more general context for the NHL problem of our study, we review in the remaining of this chapter a number of scientific publications in sports scheduling.

2.3 Literature on time-constrained sports scheduling problems

By far, most of the studies on sports scheduling either deals with leagues that play some time-constrained RRT as their regular-season schedules, or addresses a number of fundamental questions regarding such well-structured tournaments. Although the whole structure of the NHL regular season does not induce a compact schedule, the literature on both practical and theoretical aspects do provide us with insights into solving the NHL scheduling problem.

In line with the most fundamental aspects, Easton *et al.* (2004) highlight the direct analogy between latin squares and single RRT (and thus mirrored double RRT). A latin square of order n is an $n \times n$ array filled with the elements from the set $S = \{1, 2, \ldots, n\}$, such that each element appears exactly once in each row and exactly once in each column. Reductions of part of the extensive studies on latin squares to single RRT scheduling are presented in Easton (2003), providing some important results with regard to, e.g., the scheduling of tournaments with fixed games, which is one of the most common constraints found in practical applications (Easton *et al.*, 2004). Specifically, Easton (2003) uses the NP-completeness results on partially completed latin squares in Easton and Gary Parker (2001) to show that a single RRT with

Table 2.2 *Length of road trips in official NHL schedules* Frequency of road trips by their individual number of games ($\#g$) in the official schedules of past seasons

$\#g$	2009-10		2010-11		2011-12		2012-13		2013-14	
1	253	(47.83%)	262	(50.38%)	255	(47.93%)	165	(46.35%)	295	(48.60%)
2	132	(24.95%)	133	(25.58%)	130	(24.44%)	96	(26.97%)	131	(21.58%)
3	77	(14.56%)	73	(14.04%)	77	(14.47%)	50	(14.04%)	93	(15.32%)
4	35	(6.62%)	29	(5.58%)	45	(8.46%)	24	(6.74%)	59	(9.72%)
5	18	(3.40%)	18	(3.46%)	15	(2.82%)	13	(3.65%)	20	(3.29%)
6	11	(2.08%)	4	(0.77%)	7	(1.32%)	6	(1.69%)	6	(0.99%)
7	2	(0.38%)	1	(0.19%)	2	(0.38%)	1	(0.28%)	2	(0.33%)
8	1	(0.19%)							1	(0.16%)
9					1	(0.19%)	1	(0.28%)		

fixed games can be scheduled in polynomial time only for a very few special cases, as the problem is NP-complete even *i*) if on average each team has two games scheduled; or *ii*) if all but three rounds are completely scheduled and every team has at most three unscheduled games.

A latin square of order n, with n even, is closely related to 1-factorization of a complete graph K_n, which is, in turn, equivalent to an edge coloring of K_n and also to a decomposition of K_n into perfect matchings. Therefore, another analogy (probably more popular) occurs between an RRT and an edge coloring of K_n with $n-1$ colors (Easton *et al.*, 2004). To be more precise, an *oriented coloring* of the graph K_n, which is defined by a 1-factorization of K_n together with an orientation, completely defines a schedule for the single RRT: each node k in the graph correspond to a team $k \in \{1, 2, \ldots, n\}$ in the league and an arc (i, j) represents the game to be played by team i against team j, at the venue of team j, on the specific round represented by each 1-factor (and thus by each color). Several research studies focus on graph-based models, including de Werra (1980, 1981, 1982, 1985, 1988), Knust and von Thaden (2006), Schreuder (1980, 1992) and Van Weert and Schreuder (1998).

Although Combinatorics has efficient methods for some special situations (Easton, 2003; Hamiez and Hao, 2004), scheduling an RRT normally becomes a very hard combinatorial problem when additional constraints, such as arena availability or travel distances, are considered. Furthermore, researches in general have provided compelling evidence that developing appropriate models in sports scheduling is as much of an issue as the choice of solution methodologies to be employed (Trick, 2005; Kendall *et al.*, 2010) .

In particular, traditional approaches often decompose the problem into at least the following two phases:

(i) determining the opponent of every team in each round; and

(ii) determining which team in every match plays at home (so that its opponent is the one to play away) in each round.

Such phases have been sequentially solved both in the order we present them as well as in the reverse order. Indeed, Knust (2014) lists 12 references (e.g., Trick, 2001; Elf *et al.*, 2003; Miyashiro and Matsui, 2005; Post and Woeginger, 2006; Brouwer *et al.*, 2008; Cheung, 2008) that uses a *first-schedule-then-break (FSTB) approach*, which first deals with phase (i), and then, with phase (ii); and 20 references (e.g., Schreuder, 1992; Nemhauser and Trick, 1998; Henz, 2001; Miyashiro *et al.*, 2003; Rasmussen and Trick, 2007; Briskorn, 2008a; Knust and Lücking, 2009; Larson and Johansson, 2014) that uses a *first-break-then-schedule (FBTS) approach*, which proceed from phase (ii) to phase (i), instead. In those terms, a "break" refers to the occurrence of a round when a team shifts to a an away game after having played at home in the previous round, or the team shifts to a home game after an away game.

Decomposition approaches like that have benefited from fundamental progress in the design of models and algorithms for some specially-created sports scheduling problems. In general, triggered by real-world applications, such rather theoretical problems gave rise to most of the current literature on sports scheduling, which normally does not take into account many of the complex issues that appear in practice. In particular, the problems are often easy to state, either by some mathematical formulation (specifying a few types of constraints and, more often than not, an objective function) or by no more than a descriptive definition. This allows researchers to focus not only on the development of appropriate models but also in the improvement of solution methodologies which, in turn, may provide inspiration for dealing with practical problems, including others than those in sports scheduling.

Throughout the following subsections, we present an informal description of some of the most investigated problems in the sports scheduling literature, and we also point out a number of contributions to each of them. In particular, the examples we mention here include both studies that focus "only" on constructing a feasible schedule and studies that try to find the best schedule with regard to a certain evaluation function.

2.3.1 Finding home-away patterns

A *home-away pattern* (HAP) is a sequence of home games, away games, and byes related to a particular team and according to which the team plays during the tournament. Such a pattern is often represented by a vector with an entry for each round containing either an H, an A, or a B to indicate that the team has, in the corresponding round, a home game, an away game, or a bye, respectively. Regarding the construction of a schedule for n teams, an *HAP set* is a set of exactly n home-away patterns, each one associated with a particular team.

Obviously, creating an HAP set in order to have a corresponding RRT schedule requires to satisfy certain constraints (enforcing, for example, a proper pairing of the patterns). An HAP set for which an RRT schedule exists is said to be *feasible*. And the task of determining whether a given HAP set is feasible is known as the *HAP Set Feasibility Problem* (Briskorn, 2008a). Researches have been conducted to establish necessary and sufficient conditions for the feasibility of a given HAP set to different types of tournaments (de Werra, 1980, 1988; Schreuder, 1992; Van Weert and Schreuder, 1998; Easton *et al.*, 2001; Miyashiro *et al.*, 2003; Lim *et al.*, 2006; Briskorn, 2008b). However, sufficient conditions for general HAP set feasibility remain unknown.

2.3.2 Optimizing breaks

The number of consecutive games played away (at home likewise) by any given team is strongly related to fairness among all the teams in a tournament (de Werra, 1981) and also to attractiveness of the games (Bartsch *et al.*, 2006; Drexl and Knust, 2007). Therefore, the lengths of both sequences of Hs and sequences of As in an HAP is especially relevant in virtually any sports scheduling problem. When a team plays either two consecutive games at home or two consecutive games away, that team is said to have a *break* in the last of the corresponding two rounds. Ideally, the course of games played by any given team would be alternated between home and away games as regularly as possible.

Minimizing breaks is the first objective in sports scheduling (Kendall *et al.*, 2010). Lower bounds on the number of breaks have been stated, notably by de Werra (1981), for different kinds of tournaments. In particular, while it is possible to construct an RRT without any break for an odd number of teams, an RRT for n teams has at least $n-2$ breaks if n is even. Theoretical results and efficient (polynomial) methods on generating an RRT schedule with minimum number of breaks are well known when no additional constraints are considered (de Werra, 1981, 1988; Schreuder, 1992; Van Weert and Schreuder, 1998).

However, no efficient method is currently known when a cost is associated to each assignment of a game to a round and the goal is to find a schedule having that minimum number of breaks while minimizing the sum of the costs of the assignments (Briskorn, 2008b). For this special case, Briskorn and Drexl (2007) develop and apply, at first, a branch-and-price approach (with slightly disappointing results), and then, a heuristic variant of the same approach that is reported to provide good solutions for random instances (yet limited to no more than 10 teams).

In virtually all practical applications, schedules with the minimum number of breaks we just mentioned (i.e., $n-2$ for n teams) normally does not satisfy many kinds of requests. Examples of such requests are presented by Nemhauser and Trick (1998) for the basketball competition of the Atlantic Cost Conference (ACC), which includes restrictions on the place where certain teams can play on a given round and restrictions on the order in which specific opponents are visited by a team. Inspired by the works of Russell and Leung (1994) and of Schreuder (1992), they solved the scheduling problem of the ACC through a combination of IP and complete enumeration. In fact, Nemhauser and Trick (1998) used a rather typical decomposition approach that at first generates feasible home-away (and *bye*) patterns by enumeration; after, uses IP to find pattern sets for "placeholder teams"; next, uses also IP to find timetables for placeholders; and then, assigns actual teams to placeholders by complete enumeration. For each one of these four phases, a certain part of the constraints of the problem was taken into consideration. Shortly later, a similar decomposition approach

was used by Henz (2001) with the crucial difference that all the phases were then solved only through CP. Comparing to the results from Nemhauser and Trick (1998) for the ACC 1997-98 season, in which around 24 hours were needed to create all the feasible schedules, this change in the approach led Henz (2001) to obtain the same schedules in less than one minute. Almost all this drastic reduction in solution time was achieved in the last phase alone, when assigning teams to placeholders. In order to solve the same problem, as well as other variants of it with break minimization, Rasmussen and Trick (2007) proposed an algorithm that iterates between those four phases where, for a certain iteration, only a limited number of the patterns are generated and Benders cuts arising from infeasibilities identified in other phases are used to generate new patterns. The algorithm was applied to problems on both mirrored and non-mirrored schedules, with and without place constraints, and is reported to excel in performance compared to previous approaches. Later on, Rasmussen (2008) also succeed in scheduling soccer games for a triple RRT of the Danish Football Association through the same logic-based Benders approach.

Among other sports-scheduling applications involving minimization of breaks, we mention the construction of a double RRT schedule for the "Serie A" of the Italian Major Football League by Della Croce and Oliveri (2006). They report to be able to generate several schedules by adapting the decomposition approach of Nemhauser and Trick (1998), which, in the first phase, generates an HAP set that satisfies several constraints (some of them related to cable television requirements); in the second phase, produces a corresponding RRT schedule with placeholder teams; and, in the third phase, assigns actual teams to the placeholders in that schedule. Also, Van Hentenryck and Vergados (2005) propose a simulated annealing algorithm that is reported to find optimal solutions very quickly for large instances (e.g., 28 teams).

Related to this context, many studies involve finding a schedule that minimizes the number of breaks for the special situation in which, on every round, the opponent for each team is already known. Namely, given an *opponent schedule*, which is a timetable with the rows individually associated to the teams, and the columns to the rounds, such that every entry specifies the opponent of the respective team on the corresponding round, the *Break Minimization Problem* (which we refer to as BMP) consists in finding an associated feasible HAP set that results in a schedule with the minimum number of breaks. Some authors (e.g., Trick, 2011) uses the acronym CBMP to refer to a more general constrained problem where the minimization of breaks is not necessarily subject to a certain opponent schedule, but to any kind of constraint instead.

The BMP has been widely addressed in sports scheduling papers. Whether embedded in more general models or in practical applications, it normally appears in the two-phase

decomposition approach we mentioned earlier (p. 14), in which all the games (matching of teams) are assigned in one phase (*i*), while the place for each game is only determined in the other phase (*ii*).

An extensive study about the BMP (and also about some of its variations, which include new constraints) is conducted by Régin (2001) through the use of a CP approach that involves global constraints with which efficient filtering algorithms are associated. He was able to solve instances of up to 20 teams. Based on the discoveries of that study, Trick (2001) presents an IP model for the BMP that is reported to be at least competitive with Régin's CP model. He was able to solve instances of up to 22 teams. Elf *et al.* (2003) transform the BMP into a maximum cut problem in an undirected graph and then apply a branch-and-cut algorithm to both randomly-generated instances and the real-world instance of the Bundesliga 1999-2000 (the first national German soccer league). They were able to solve instances of up to 26 teams. Miyashiro and Matsui (2006) and Suzuka *et al.* (2007) formulate the problem as a maximum restricted cut and a maximum 2-satisfiability problems, and apply the approximation algorithm by Goemans and Williamson (1995), which is based on a positive semidefinite programming relaxation. They were able to solve instances of up to 40 teams.

Despite the advantages of having a schedule with minimum number of breaks, longer sequences of consecutive away games would imply fewer road trips for a team and thus reduced traveling distances in total. Therefore, in some contexts, normally where venues are located far from each other (as in the NHL problem), it is preferable to have a large number of breaks, so as to minimize traveling distances. Russell and Leung (1994) are the first to consider the problem of finding a HAP set that maximizes the number of breaks for a given opponent schedule. But breaks minimization and breaks maximization were first treated together by Miyashiro and Matsui (2005). In particular, these two problems are shown to be equivalent for a single RRT if the opponent schedule is known beforehand: an optimal solution for one problem can be directly constructed from an optimal solution for the other one.

Urrutia and Ribeiro (2006) and Rasmussen and Trick (2007) deal with maximization of breaks in special cases of the Traveling Tournament Problem, which we describe in the following subsection.

2.3.3 The Traveling Tournament Problem

The duration of a tournament might be too short for teams to play according to the highly-alternating HAP of a schedule with only a few breaks. This is especially the case when, as in the NHL, there are venues located very far from each other. Due to obvious

economic issues and possibly to players' fatigue (Smith *et al.*, 2000), only trying to alternate home games and away games as much as possible may not be suitable for some sports leagues around the world (Oberhofer *et al.*, 2010). In fact, in order to reduce traveling distances (and times), a team should have as many consecutive away games as possible, considering it goes directly from one opponent's venue to that of the next one without returning home before the corresponding breaks.

Inspired by a real-world sports scheduling problem that requires minimization of traveling distances, namely the scheduling of the Major League Baseball (MLB) regular season, Easton *et al.* (2001) introduces the *Traveling Tournament Problem* (TTP). Given the distances between each pair of venues, a lower bound l and an upper bound u on the number of consecutive home and consecutive away games, the TTP consists in scheduling a double RRT for n teams that minimizes the total distance traveled during the tournament by all teams, subject to the constraint that every team must play between l and u consecutive away games and between l and u consecutive home games. Typical values for l and u are 1 and 3, respectively. As noted by Trick (2011), most of the studies on the TTP also include the so-called *no-repeater constraint*, which imposes that if a team i plays at home against a team j in one round then j must not play at home against i in the next round. Obviously, as in the NHL problem of this thesis, every team on a TTP is assumed to have its own venue at its home city, where it begins and ends the travels over all the tournament. In addition, every time a team plays two consecutive away games, it travels directly from the venue of its first opponent to that of the second one.

Since it was first announced, in 2001, some variants of the TTP have been proposed in order to capture further requirements in sports scheduling applications, and also, to stimulate researchers on the development of new approaches. Among such variants, we mention the *Timetable Constrained Distance Minimization Problem* (TCDMP), introduced by Rasmussen and Trick (2006), and the *TTP with Predefined Venues* (TTPPV), proposed by Melo *et al.* (2007). The TCDMP can be defined as follows: given an opponent schedule (timetable) for a double RRT, the distances between each pair of venues, and a lower bound and an upper bound on the number of consecutive away games and consecutive home games, find a corresponding feasible HAP set which minimizes the total distance traveled during the tournament by all teams. Therefore, the TCDMP is a generalization of the BMP, which we defined earlier (p. 17), where distances are now considered in the objective function, instead of breaks (Rasmussen and Trick, 2009). As for the TTPPV, it can be defined as follows: given the venue where each game of a single RRT will take place and the distances between any pair of venues, find a schedule in which the total distance traveled by all teams is minimized and no team plays more than three consecutive away games or more than three consecutive

home games.

The TTP and its variants are among the most studied problems in the sports scheduling literature. Indeed, the classification of pertinent literature by Knust (2014) currently list almost 50 papers that deals with the TTP. Numerous benchmark instances and best-found results are presented on the TTP website `http://mat.gsia.cmu.edu/TOURN/`. In particular, the "NL instances" introduced by (Easton *et al.*, 2001) were inspired in the National League (one of the two leagues constituting the MLB), and today, are probably the best-know class of TTP instances. Although the TTP is very simple to state and typical instances are made up from quite little data, it has proved to be very difficult to solve. Even for a very small number of teams, TTP instances have challenged many researchers in combinatorial optimization. In fact, in its original form, only instances with as few as 10 teams have been solved to optimality. Curiously, for example, the classic NL8 instance (with 8 teams) was only solved to provable optimality for the first time by Irnich (2010), almost 10 years after it was first announced.

Several state-of-the-art algorithms have been systematically applied in order to determine either improved lower bounds or better objective function values for benchmark instances of the TTP. The first algorithms, which combine CP and IP approaches, were proposed by Easton *et al.* (2001, 2003) and Easton (2003). They were able to find an optimal solution for the NL8 without the non-repeater constraint by means of a parallel implementation of a branch-and-price algorithm in which an IP approach solves the master problem and a CP approach solves the pricing problem. Later on, Irnich (2010) revisited that CP-based branch-and-price, reformulated its pricing problem as a shortest-path problem over an extended network, and implemented new branching techniques, which led him to obtain several improved lower bounds and some optimal solutions for previously unsolved TTP instances, including the NL8 (with the non-repeater constraint). Around the same time, Uthus *et al.* (2009) presented a remarkable work in which a parallelized DFS* (a variant of a depth-first branch-and-bound search that keeps all heuristic estimates in memory) is able to find past known optimal solutions much faster than previous approaches and to greatly improve the lower bounds of larger TTP instances. More recently, those same authors (Uthus *et al.*, 2012) also proposed a new approach based on iterative-deepening A* (a greedy best-first-search guided by a heuristic strategy) that is able to find past known optimal solutions even faster than their DFS* and that turns out to be the first approach to find optimal solutions to all 10-team instances (including the NL10) of the four TTP classes to which it was applied.

2.3.4 Typical constraints in sports scheduling

Although each sports scheduling problem has its own set of constraints and objectives, the literature often come up with common requirements found in different studies. In 2010, on the occasion of the International MultiConference of Engineers and Computer Scientists, a dozen of authors (Nurmi *et al.*, 2010) proposed a framework for sports scheduling problems as an attempt to establish a common ground for the development of benchmark instances which would allow researchers to evaluate and compare their solution approaches. In particular, Nurmi *et al.* (2010) outline 36 typical constraints inspired from both theoretical and real-world problems. For the sake of clarity, the statements for those constraints are reproduced bellow.

— Structural requirements:

C01. There are at most R rounds available for the tournament.

C02. A maximum of m games can be assigned to round r.

C03. Each team plays at least m_1 and at most m_2 games at home.

C22. Two teams play against each other at home and in turn away in 3RR or more.

— Home-away requirements:

C04. Team t cannot play at home in round r.

C05. Team t cannot play away in round r.

C06. Team t cannot play at all in round r.

C07. There should be at least m_1 and at most m_2 home games for teams $t_1, t_2, \ldots$ on the same day.

C08. Team t cannot play at home on two consecutive calendar days.

C09. Team t wants to play at least m_1 and at most m_2 away tours on two consecutive calendar days.

C23. Team t wishes to play at least m_1 and at most m_2 home games on $weekday_1$, $m_3 - m_4$ on $weekday_2$ and so on.

— Special-game requirements:

C10. Game h-team against a-team must be preassigned to round r.

C11. Game h-team against a-team must not be assigned to round r.

C24. Game h-team against a-team cannot be played before round r.

C25. Game h-team against a-team cannot be played after round r.

C34. Game h-team against a-team can only be carried out in a subset of rounds $r_1, r_2, r_3, \ldots$.

— Break requirements:

C12. A break cannot occur in round r.

C13. Teams cannot have more than k consecutive home games.

C14. Teams cannot have more than k consecutive away games.

C15. The total number of breaks must not be larger than k.

C16. The total number of breaks per team must not be larger than k.

C17. Every team must have an even number of breaks.

C18. Every team must have exactly k number of breaks.

C35. A break of type A/H for team t_1 must occur between rounds r_1 and r_2.

— Tournament-quality requirements:

C19. There must be at least k rounds between two games with the same opponents.

C20. There must be at most k rounds between two games with the same opponents.

C21. There must be at least k rounds between two games involving team t_1 and any team from the subset $t_2, t_3, \ldots$.

C26. The difference between the number of played home and away games for each team must not be larger than k in any stage of the tournament (a k-balanced schedule).

C27. The difference in the number of played games between the teams must not be larger than k in any stage of the tournament (in a relaxed schedule).

C36. The carry-over effects value must not be larger than c.

— Strength-group requirements:

C28. Teams should not play more than k consecutive games against opponents in the same strength group.

C29. Teams should not play more than k consecutive games against opponents in the strength group s.

C30. At most m teams in strength group s should have a home game in round r.

C31. There should be at most m games between the teams in strength group s between rounds r_1 and r_2.

C32. Team t should play at least m_1 and at most m_2 home games against opponents in strength group s between rounds $r_1$and r_2.

C33. Team t should play at least m_1 and at most m_2 games against opponents in strength group s between rounds r_1 and r_2.

In general, a benchmark instance for a highly-constrained sports scheduling problem can then be characterized by standard information, which includes the number of teams, the type of RRT, and among the constraints from the list above, the sets of both hard and soft constraints taken into account, with their respective parameters. Obviously, these 36 constraints

do not cover all the constraints that might appear in many real-world sports scheduling problems. In particular, they are not inclusive enough to allow a complete specification of an instance for the NHL scheduling problem, as we will point out in the next chapter.

In their survey on round-robin scheduling, Rasmussen and Trick (2008) characterize the following eight common types of constraints, for which we identify some possible relations to the constraints in Nurmi *et al.* (2010).

— *Place constraints* Constraints enforcing certain teams to play at home or away on specific rounds. They arise, for example, from unavailability of venues, and in general, can be specified by C04.

— *Top-team and bottom-team constraints* Constraints imposing that teams from a similar strength group must have their games somewhat evenly distributed through all the rounds, or that certain teams should not play consecutive games against opponents from a similar strength group. They can usually be specified by the statements from C28 to C33.

— *Break constraints* Constraints ensuring that no breaks occur on certain rounds. They can be stated by C12.

— *Game constraints* Constraints fixing or forbidding certain games on particular rounds. They can be specified by C10 or C11.

— *Complementary constraints* Constraints imposing that certain teams must not be assigned to home games that would be simultaneously played on any particular round. They arise, for example, when two teams share the same venue as home. Although Nurmi *et al.* (2010) does not directly specify any particular constraint for this case, such constraints could be stated by C31 with a "strength group" s being the set of corresponding teams, $m = 1$, and r_1 and r_2 being the first and the last rounds, respectively.

— *Geographical constraints* Constraints enforcing the games to be, on any round, somewhat evenly distributed throughout all regions of the tournament. They could also be stated by C31 with a "strength group" s being the set of teams with venues in a specific region, m being a reasonable number of games, and r_1 and r_2 being the first and the last rounds, respectively.

— *Pattern constraints* Constraints over the HAP of each team ensuring, for example, that the number of consecutive games at home (or away) stays within a certain range, or yet that all teams have the same number of breaks in their HAPs. Such constraints can usually be specified by the statement from C12 to C18, and by C35.

— *Separation constraints* Constraints imposing lower and upper bounds on the number of rounds between consecutive games involving the same pair of teams. Obviously, they arise from an attempt to evenly spread such games through the whole period of the tournament, and in general, can be specified by the statements from C19 to C21.

2.4 Literature on time-relaxed sports scheduling problems

Nearly all the scientific literature on sports scheduling has focused on the study of compact schedules. This predominance of researches on time-constrained problems may be due to both the rather neat structure of a compact schedule and their high popularity among major professional sports around the world.

In non-commercial leagues, however, time-relaxed schedule is the most widely adopted structure (Knust, 2010). The regular seasons of both the NHL and the National Basketball Association (NBA) are two rare examples of major tournaments in which the number of days "available" for potential assignments of games is much larger than the number of games to be played by a team. In particular, teams may play different numbers of games during a certain week, and the number of weekly games are often unevenly spread throughout the season.

Scheduling for non-commercial leagues normally requires special attention not only to the limited access to sports facilities for the home games of each team, but also to the sporadic availability of sportsmen. In fact, in such contexts, a team is usually able to provide only a relatively small number of potential dates for its home games, while many other dates may not be available for away games, either.

In a recent work, Knust (2010) deals with non-commercial sports league scheduling. A time-relaxed schedule is to be found for a double RRT with several hard constraints, including unavailability of teams to play away on certain dates, and also soft constraints, including minimization of breaks. In particular, the tournament is assumed to be made up of two half series, one after the other, and every team must play once at home in one series and once away in the other series against each other team. Either half series refers thus to a single RRT. In addition, regarding each team, the number of home games must not differ by more than one from the number of away games. This restriction over a single RRT defines a *balanced home-away assignment* for the matches (Knust and von Thaden, 2006). The paper presents two formulations for the problem, one as an IP model and the other as a multi-mode resource-constrained project scheduling problem, and a two-phase heuristic algorithm is proposed. In the first phase of the algorithm, a balanced home-away assignment for the matches is determined with an implementation of connected neighborhood structures introduced with theoretical results by Knust and von Thaden (2006). In the second phase, an adapted genetic

algorithm for resource-constrained project scheduling problems is used, and then, matches are reassigned to different days as an attempt to evenly distribute the games over the whole season. The heuristic is reported to be very efficient for a number of benchmark instances and was then used to schedule two seasons of regional table-tennis leagues in Germany. Knust (2010) mentions trying to solve, with a similar two-phase approach, time-constrained problems subject to a hard constraint enforcing the number of breaks to be at their minimum, but feasible solutions were only found for a few instances. This might indicate that the approach is not well suited for the special structure of time-constrained problems, even so the minimum-break requirement seems too restrictive.

For other studies on scheduling non-commercial leagues, we refer to Schönberger *et al.* (2000) and to Schönberger *et al.* (2004), where similar tournaments (also for a table-tennis league in Germany) are scheduled by Genetic Algorithms and by CP approaches.

Another example of real-world tournament that gives rise to a time-relaxed schedule is the 1992 World Cup of Cricket. Addressed by Armstrong and Willis (1993), the problem consists in scheduling a single RRT for 9 teams, such that every team plays its corresponding 8 games in a period of 26 days, and for which 19 venues scattered over a large geographical area (in Australia and New Zealand) are available. Among the many constraints taken into account, some are related to long traveling distances and others to several requirements from TV broadcasting. The paper presents an IP formulation, but due to both its large number of constraints (which turned out to be impractical for that time) and the need for having a more flexible approach, heuristic procedures are proposed to solve the problem. Implemented in a spreadsheet package (Lotus 1-2-3), such heuristic procedures allow a user to sequentially assign games to days in an interactive manner.

It seems that no other sport scheduling in the literature share so many identical issues with the NHL scheduling of this thesis as that of the NBA regular season. Therefore, we devote the following subsection specifically to studies on the scheduling of games for the NBA.

2.4.1 The NBA scheduling problem

Like the NHL, the NBA currently consists of 30 teams, each one playing 82 games (41 at home and 41 away) according to a relaxed schedule for its regular season. The NBA features two 15-team conferences of three 5-team divisions each. Also, the teams are scattered throughout a vast geographical area in North America, 29 of them in the United States and one in Canada.

To our knowledge, the problem of scheduling the NBA regular season has been studied at least on two occasions: in 1980 in a paper by Bean and Birge, and in 2009 in a Ph.D. thesis

by Bao. Generally speaking, the approach by Bean and Birge (1980) is quite practical and had, in fact, been applied to actual instances of at least four NBA regular seasons. It has probably served as an important baseline for the approach that Fleurent (1987) proposed later on for the NHL scheduling problem. The study by Bao (2009), on the other hand, follows an approach that is more in line with the researches on time-constrained problems that we mentioned earlier. He points out several types of constraints for the NBA scheduling problem, and for each type, he proposes both IP and CP formulations, which are implemented into a commercial solver and individually tested on randomly-generated instances. The remaining of this subsection is our review on these two studies.

Bean and Birge (1980) develop schedules for some seasons in the 1980's, when the NBA consisted of only 22 teams but each team had already to play 82 games in a period of around 170 days. The NBA scheduling problem of that time was to assign 902 games to dates of the regular season, such that a number of constraints were satisfied and the total traveling distance was minimized. In their paper, they mention that a typical arena used by an NBA team was usually available only on about 30% of the days of the season, and that this turned out to be the most difficult constraint they had to deal with.

To provide some insight about the size and complexity of the problem, Bean and Birge (1980) present a mathematical formulation of a modified problem in which the season would consist of exactly 82 days and the teams would play every day of this fictional season. Of course, a solution for the problem would now be a compact schedule. In their model, the constraints are linear equalities and the objective function is also linear, all of them expressed in terms of four-index variables $x_{ijkl} \in \{0, 1\}$ for any triple of teams i, j, k, and every day l. One such variable, x_{ijkl}, is set to value 1 only if team i is assigned to travel from the city of team j to the city of team k to play a game on day l. Clearly, the model has then $O(n^4)$ variables. The objective function, whose value is to be minimized, is the sum of the daily traveling distances of all the teams, and the constraints impose the following structural requirements: *i*) for each pair of teams, a specific number of games must be played during the season; *ii*) if a visiting team is in a city to play, the respective home team must also be there; *iii*) all teams start the season at their own city and half of them go to visit the others in the first day; and *iv*) if a team leaves one city then it must have played in that city on the preceding day. Regarding the seasons in the 1980's, the model entails almost 42 thousand constraints and more than 800 thousand variables. For the current seasons, where 30 teams play 82 games per team, it would have almost 78 thousand constraints and more than 2 million variables. Bean and Birge (1980) claim thus that finding a solution even for such simplified unrealistic formulation would be extremely difficult at the time.

They propose then a two-phase heuristic approach for the NBA scheduling problem. In

the first phase, road trips with up to five away games for each team are created, and in the second phase, those trips are combined into a feasible schedule for the entire league. To be more precise, the approach can be outlined as follows.

— *Phase I: Generating road trips* This phase is similar to a classic savings heuristic originally proposed for routing problems by Clarke and Wright (1964). In particular, the following saving measure is used, for each team k, to select the opponents, i and j, to be visited during a same road trip for k:

$$s_k(i,j) = \mathsf{dist}(i,k) + \mathsf{dist}(k,j) - \mathsf{dist}(i,j)$$

where $\mathsf{dist}(\cdot,\cdot)$ is the distance between the cities of the respective pair of teams. The feasible road trips of each team k are generated by first creating every road trip, $r = (k,l,k)$, made up of a single away game against team l, and then, by repeating the following steps, where a feasible road trip is always limited to at most five games. *Step 1)* Calculate the saving, $s_k(i,j)$, for each pair of teams, i and j. *Step 2)* Create a list with all (i,j)-pairs sorted in a non-increasing order of their savings. *Step 3)* Make a single pass through that ordered list and for each pair at hand, (i,j), merge the road trip that contains one of the corresponding teams, i, with the road trip that contains the other team, j, into a new feasible road trip if no more than five games are involved and if one team is at the end of one trip and the other team is at the begin of other trip, i.e., if the forms of those two trips are either $(k, \ldots, i, k)$ and $(k, j, \ldots, k)$, or $(k, \ldots, j, k)$ and $(k, i, \ldots, k)$.

— *Phase II: Scheduling road trips* In this phase, the heuristic firstly sorts all road trips by their total traveling distances, and subsequently, try to schedule them, from the longest to the shortest ones, into periods in which the corresponding traveling team has the least arena-available dates. When the road trip at hand cannot be feasibly scheduled, it is divided into two or more partial road trips, which in turn are to be scheduled with the same "sort-and-schedule" strategy, but only after all the current road trips have been considered. This iterative process continues until only road trips made up of a single game remain unscheduled. The games in such road trips are then individually scheduled to a feasible day that minimizes the increase in overall traveling distance.

This two-phase heuristic was adapted later on by Fleurent (1987) for the NHL scheduling problem. This shall become evident to the reader in Chapter 5, especially when we describe the "forced-trip heuristic" and the "free-trip heuristic", which generates road trips with games to be played far from the city of the corresponding traveling team.

Bean and Birge (1980) report that their heuristic alone was able to find feasible schedules

for the NBA games. But since optimality was not guaranteed, they mention to have eventually used "switching algorithms" in an attempt to further reduce the overall traveling distance. Such algorithms would find the best feasible dates for each game, while considering fixed the rest of the schedule. They are, however, reported to lead to only few improvements over the initial feasible schedule. In spite of its rather vague description, it appears that this strategy was similar to the "exchange heuristics" used later on by Fleurent (1987) (which is described in Ferland and Fleurent, 1991, as well).

More recently, Bao (2009) has also tackled the NBA scheduling problem in a chapter of his Ph.D. thesis. He claims that a schedule for the NBA regular season must satisfy numerous constraints, and then, he presents several of them. In particular, each constraint is formulated both by means of Integer Programming and by means of Constraint Programming. Computational results obtained with IBM ILOG CPLEX® 11.2 and with IBM ILOG CP Optimizer® 2.1 (for the IP and the CP models, respectively) are compared between them with regard to running time, as well as number of branch nodes (from the IP solver) and number of failed branches (from the CP solver).

The experiments have been individually conducted for each type of constraints added to a basic formulation that only considers the structural requirements of the problem. The instances submitted to the solvers have been generated for even numbers of teams varying from 6 to 30. In each instance, the number of days provided by Bao (2009) is equal to the double of the corresponding number of games that every team has to play. The following is our informal description of those experiments, which, to provide an idea of how challenging the problem turned out to be under each type of constraint, also includes some of the main reported results. Bao (2009) limits each experiment to 30 minutes and uses a personal computer running Windows® XP with 2.2GHz Duo Core CPU and 1GB of RAM.

— *Structure* A single RRT is to be scheduled, such that only the essential constraints for such a tournament are taken into account, i.e., every team plays each of the others exactly once and no team plays more than one game per day. Feasible solutions for all the instances with up to 30 teams are found both by IP (in as long as 2.56 seconds) and by CP (in no more than 0.76 second).

— *Interconference games* A double RRT is to the scheduled for the interconference games, which are evenly divided into two conferences, such that every team plays once at home against each opponent. In the NBA, the structure of the interconference games is indeed a double RRT. While IP finds feasible solutions for the instances with up to 30 teams (in no more than 22 seconds), CP solves only instances with up to 22 teams (in less than 4 minutes), failing to solve the instances with 26 or more teams (in 30 minutes).

— *Intradivision games* A quadruple RRT is to be scheduled, such that every team plays twice at home against each opponent. In the NBA, the structure of the intradivision games is indeed a quadruple RRT for each division (which currently involves five teams). Only instances with up to 10 teams were submitted to the solvers, and both IP and CP easily find feasible solutions (in no more than 0.17 second).

— *Consecutive games* A single RRT is to be scheduled, such that: no team plays at home on two consecutive days; no team plays more than two games on three consecutive days; and no team plays more than five games on eight consecutive days. Feasible solutions for all instances with up to 30 teams were found both by IP (in less than 3 seconds) and by CP (in as long as 2.3 minutes).

— *Consecutive byes* A single RRT is to be scheduled, such that: any team plays at least two games per week and at least one game during every five days. Feasible solutions for all instances with up to 30 teams were found both by IP (in as long as 4 minutes) and by CP (in less than 20 seconds).

— *TV schedules* A single RRT is to be scheduled for instances in which about 20% of the games are considered as being "attractive games", such that: no more than a given number of attractive games are played on a same day; at least a certain number of attractive games are played on each of the days in a given special set of days; and every team plays at least two games per week. Feasible solutions for all instances with up to 30 teams were found both by IP (in no more than 0.41 second) and by CP (in no more than 0.91 second).

— *HAP sets* A single RRT is to be scheduled according to a given (feasible) HAP set. While IP finds feasible solutions for instances with up to 30 teams (in no more than 8.5 minutes), CP solves only instances with up to 14 teams (in less than 0.03 second) and fails to solve the instances with 18 or more teams (in 30 minutes).

— *Arena availability* A single RRT is to be scheduled for instances in which the arena of any team is available only on about 33% of the days. While IP finds feasible solutions for instances with up to 30 teams (in less than 2.3 seconds), CP solves only instances with up to 14 teams (in as long as 27 seconds), failing to solve the instances with 18 or more teams (in 30 minutes).

— *Forbidden assignments* A single RRT is to be scheduled for instances where, for each day, certain particular games should not be assigned. In the NBA, this occurs when, for example, a team requests not to have certain games on a particular day. Feasible solutions for all instances with up to 30 teams were found both by IP (in no more than 0.63 second) and by CP (in no more than 1.08 seconds).

— *Complementary home games* A single RRT for up to 30 teams is to be scheduled, such that exactly two given teams never play at home on a same day. In the NBA, this constraint occurs because two teams (the LA Lakers and the LA Clippers) share the same arena for their home games. Feasible solutions for all instances with up to 30 teams were found both by IP (in no more than 0.03 second) and by CP (in as long as 1.17 seconds).

— *Assignment-value maximization* A single RRT is to be scheduled for instances where a fixed value (between 0 and n^2 for a problem with n teams) is given for each potential game-day assignment, and the sum of the values for the actual assignments in a feasible solution is to be maximized. While IP finds optimal solutions for instances with up to 30 teams (in less than 2 seconds), CP fails to solve even the instances with only 6 teams (in 30 minutes). This contrasting result is somehow expected, since the solvers are dealing with an optimization problem in this particular experiment.

— *Back-to-back games* A single RRT is to be scheduled, such that every team plays a same given small number of back-to-back games. Only instances with up to 10 teams were submitted to the solvers, and both IP (in no more than 34 seconds) and CP (in as long as 14 minutes) find feasible solutions.

— *Weekend games* A single RRT is to be scheduled, such that every team plays a same given small number of games on weekends (Fridays, Saturdays, and Sundays). Only instances with up to 24 teams are solved by IP (in as long as 29.6 minutes); and although solutions are found for instances with 20, 24, and 28 teams (in just 0.38 second), CP fails to solve (in 30 minutes) almost 40% of the instances with up to 30 teams.

— *Bounded traveling distances* A single RRT is to be scheduled, such that the total traveling distance for each team remains within a particular given range. Both IP and CP fail to find feasible solutions, even for those with only as few as 6 teams.

It is important to notice that all these experiments have been undertaken one at a time, which might be suitable on providing insights into the development of decomposition approaches as those we mentioned earlier for minimizing breaks. In particular, Bao (2009) does not test models integrating into a same formulation the different types of constraints.

CHAPTER 3

THE NHL SCHEDULING PROBLEM

The NHL scheduling problem consists in assigning a playable day to each game of the NHL regular season by not imposing more than one single daily game for any team. The game-day assignments should obviously satisfy several other constraints that are imposed either by the NHL, or individually, by the teams. Although the requirements, the goals, and even the structure of the NHL may change from one season to another, we will try to establish the most essential components of a good NHL schedule in the present days. In particular, all the factors taken into account in the pertinent literature that are relevant nowadays are being considered in this thesis.

In this chapter, we present the structure of the NHL and the structure of its regular season. The requirements and goals for an NHL schedule are described, and then, compared with those in the literature on the same problem.

3.1 The structure of the NHL

The NHL is currently formed by 30 franchised member clubs, which we refer to as *teams*, that are located throughout the vast territories of the United States and Canada. The teams are distributed into two *conferences* which, over the past years before 2013, has individually aggregated three *divisions* of five teams each. Since the 2013-14 season, however, the NHL teams are split into four divisions of either seven or eight teams: two eight-team divisions form the Eastern Conference, and two seven-team divisions form the Western Conference. This structure is shown in Table 3.1.

Because the NHL operates as a franchise system, the term "division" refers to a group of teams arranged not by their competitive level but by other factors, which may include their geographical locations, their rivalries, and their time zones. In particular, the teams within any specific division usually have their individual venues (for which we adopt the standard term *arenas*), located in the same geographical region. But overall, the distances between the home arenas of two NHL teams range from only a few miles to as much as 2700 miles (about 4500 km). Furthermore, comparing to the case of the Eastern Conference, the arenas in the Western Conference are, for the most part, located much farther from each other. This fact normally results in longer travel times for the Western teams, which could be somewhat aggravated or relieved, depending on the schedule of games assigned to them.

Table 3.1 *The current NHL structure* The NHL teams disposed by their respective division and conference.

Western Conference	*Central Division*		*Pacific Division*	
	CHI	Chicago Blackhawks	ANA	Anaheim Ducks
	COL	Colorado Avalanche	ARZ	Arizona Coyotes
	DAL	Dallas Stars	CGY	Calgary Flames
	MIN	Minnesota Wild	EDM	Edmonton Oilers
	NSH	Nashville Predators	LAK	Los Angeles Kings
	STL	Saint-Louis Blues	SJS	San Jose Sharks
	WPG	Winnipeg Jets	VAN	Vancouver Canucks
Eastern Conference	*Atlantic Division*		*Metropolitan Division*	
	BOS	Boston Bruins	CAR	Carolina Hurricanes
	BUF	Buffalo Sabres	CBJ	Columbus Blue Jackets
	DET	Detroit Red Wings	NJD	New Jersey Devils
	FLA	Florida Panthers	NYI	New York Islanders
	MTL	Montreal Canadiens	NYR	New York Rangers
	OTT	Ottawa Senators	PHI	Philadelphia Flyers
	TBL	Tampa Bay Lightning	PIT	Pittsburgh Penguins
	TOR	Toronto Maple Leafs	WSH	Washington Capitals

In general, changes in the NHL structure occur (when they do) on a very small scale from one year to the next. In the early 2013, however, a new NHL structure with the 30 teams distributed into only four divisions was proposed, which is now being implemented for at least three NHL regular seasons, starting at the 2013-14 season (Rosen, 2013). A method for changing sports leagues structures, which falls outside the scope of this thesis, is proposed by Macdonald and Pulleyblank (2014), specially for the case of the NHL, as an attempt to enable the construction of better schedules with regard to the total travel distance for the whole league.

3.2 The structure of the NHL regular season

The NHL regular season usually starts in the first week of October each year and runs until mid-April. During that period, a total of 1230 games are played. Each game is a match between a *home team*, which is the owner of the arena where the game is held, and an *away team* (or likewise referred to as *visiting team*). Exactly 82 regular season games are played by every NHL team, 41 at *home* and the other 41 games on the road (or equivalently, *away*).

All those games will have been completely specified by the time a schedule is designed.

For each NHL regular season between 2008 and 2013, for example, every team played 24 *intradivisional games* (against the other four teams in its own division), 40 *interdivisional games* in its own conference (against the 10 teams from the other divisions), and 18 *extra-conferencial* games (against the 15 teams from the other conference). Specifically, the intradivisional games for any team consisted of six games against each of the other four teams in its own division, where three of them were held at home, and the three others, away. With respect to the interdivisional games, a team played four times, two at home and two away, against every single extra-divisional rival in its own conference. Finally, the remaining 18 games for any team consisted of one game against each of the 15 teams in the other conference and three wild-card games versus three of those teams.

At the present, however, every team plays two games against each team outside its conference, and three games against each team inside its conference but outside its division. In addition, every team plays either four or five games against the other teams in its own division.

As mentioned earlier, the aggregation of teams into the two conferences and their different divisions is substantially related to the geographical distribution of the arenas of those teams. In particular, because teams within the same division are often located in the same region, the NHL regular season structure indicates that teams based within the same region usually play between themselves more often than those located farther from each other. Hence, most of the games give rise to trips over relatively short distances. The total distance that a team must travel, however, strongly depends on the whole schedule for the regular season.

3.2.1 The availability of dates, teams, and arenas

Even though the NHL regular season runs from early October to mid-April, there are some dates during that period of around 190 days on which no game scheduling is allowed. Indeed, several constraints are imposed either by the NHL or by any of its member clubs to the dates suitable for having a game. Firstly, the NHL managers usually specify a few dates that must be excluded from the playable dates. Such restriction often arises from some holidays or special events like the Winter Olympic Games and the NHL All-Star Game. These events alone typically shut out from consideration between seven and fourteen days at the midway point of the regular season. Secondly, it may happen that a team is not available to play at some specific date because of an exceptional event to which the club is locally committed, for example. So the individual availability of the teams may also be an important concern when designing an NHL schedule. Finally, the factor to which the pertinent scientific literature, specially Ferland and Fleurent (1991), gives the most attention is the availability of arenas.

Although every team has its corresponding arena where home games take place, the building housing an arena may also be used for a variety of other entertainments. In fact, an arena might be assumed to be unavailable for a certain day if any major event is occurring in the same city or region where a club is based on, which could inhibit interest of fans or causes public safety concerns, for example.

The availability of arena differs from one team to the other and some clubs may hardly be able to provide the minimum number of feasible dates for home games required by the NHL managers. Costa (1995) mentions the case of the 1993-94 regular season, when the number of home games per team changed to 41 and the NHL managers asked each club to provide at least 56 (thus 15 extra) arena dates for the scheduling of those games. He reports that such dates may not be enough to yield a reasonable schedule. According to Ferland and Fleurent (1991), this is the main reason why an NHL schedule design usually follows through several months of mutual discussions and arrangement of dates.

3.3 The scheduling problem of the NHL regular season

Here, we describe the NHL scheduling problem in a rather informal manner. A mathematical model with integer variables will be proposed in Chapter 4.

Because several games are played almost every day of the NHL regular season, the standard meaning of "bye" in the sports scheduling literature (Kendall *et al.*, 2010) will be used for the NHL scheduling problem. So a team is said to have a *bye* on a certain day if that team is not scheduled to play on that day.

Moreover, we define a *road trip* (or simply *trip*) for an NHL team as a sequence of all its away games scheduled to a period with no home games for that team and during which it has no more than two byes (off days) between two consecutive games. Whenever a road trip has more than one game, the corresponding travelling team plays then at least one game for each three consecutive days within that road trip period.

We denote by $a@h$ a game where a team a plays against a team h in the arena of h, and by $[a@h\colon d]$ the scheduling of the game $a@h$ on a day d, or equivalently, the *assignment* of a day d to the game $a@h$. An NHL schedule can then be seen as a set of game-day assignments for the corresponding season.

We enumerate and define below the constraints for the NHL scheduling problem that have been taken into account in this study. For each constraint, we also specify how the corresponding violations are being counted for our evaluations of NHL schedules.

In addition to the (hard) constraint that a team can never be scheduled to play more than one game a day, which we refer to as C_0, and also to the (hard) constraint that the

games must be scheduled exclusively on playable dates provided by the NHL, we consider the following nine constraints, and respective violation counting, for the NHL scheduling problem:

- C_1 *Arena availability for home games* A team should play at home only when its arena is available. Here, we count one violation for each game $a@h$ scheduled to a day d that is not an arena date provided by the home team, h.
- C_2 *Number of games over three days* A team should not play more than two games during any three consecutive days. Here, we count one violation for each sequence of three consecutive days, from a day d to day $d+2$, during which a team t is scheduled to play three games.
- C_3 *Number of games over five days* A team should not play more than three games during any five consecutive days. Here, we count one violation for each sequence of five consecutive days, from a day d to day $d+4$, during which a team t is scheduled to play more than three games.
- C_4 *Daily travel distance* A team should not travel more than 900 miles to play games on two consecutive days if those games are to be held in the arenas of two teams from different divisions. Here, we count one violation for each time a team t is scheduled to play on two consecutive days, d and $d+1$, and the respective games are to be held in two arenas (possibly including the arena of t) that are both more than 900 miles away from each other and belong to teams from different divisions.
- C_5 *Number of games in a week* A team should play at least two games in each week, from Sunday to Saturday. Here, we count one violation for each team t and each week w where t is not scheduled to play at least two games within the period from Sunday to Saturday. Weeks that involve Christmas and NHL All-Star game, or Olympic Games, are not counted.
- C_6 *Number of days between revisits* Games should be at least 14 and 30 days apart when related to the same intradivisional and interdivisional matches, respectively. Here, we count one violation for each pair of days d_1 and d_2 assigned to a match $a@h$ (same ordered *away-home* pair of teams), such that the condition $d_2 - d_1 < \delta$ holds for $\delta = 14$ if a and h belong to the same division, or for $\delta = 30$ if a and h belong to different divisions.
- C_7 *Number of games in a trip* A team should play no more than seven games during a trip. Here, we count one violation for each team t and a corresponding trip $r(t)$ during which t is scheduled to play more than seven games.

C_8 *Duration of a trip* A trip should last no more than 14 days. Here, we count one violation for each team t and a corresponding multi-game trip $r(t)$ during which t is scheduled to play a first away game on day d_1^r and a last away game on day d_2^r, and the condition $d_2^r - d_1^r \geq 14$ holds.

C_9 *Number of days between two trips* Two consecutive trips for a team should be separated by at least three days between them if that team travels more than 900 miles both from the last game in the first trip to home, and from home to the first game in the second trip; otherwise, the two trips should be separated by at least two days. Here, we count one violation for each team t and two corresponding consecutive trips, $r_1(t)$ and $r_2(t)$, where the last game in $r_1(t)$ is scheduled to be played on day d_1^r in the arena of team h_1, and the first game in $r_2(t)$ is scheduled to be played on day d_2^r in the arena of team h_2, such that $d_2^r - d_1^r \leq 3$ holds if the distance between h_1 and h_2 is farther than 900 miles, or $d_2^r - d_1^r \leq 2$ holds, otherwise.

The number of constraint violations in the last five official NHL schedules are shown in Table 3.2. Overall, these numbers seem to indicate that at least some of the constraints listed above (especially C_4, C_5, and C_6) might not be as relevant today as they had been in the past. On the other hand, these constraints are highly related to two common concerns in the sports scheduling literature, namely traveling distances and flow of the games (in an ideal situation, the games would be evenly spread throughout the whole season).

In accordance with Ferland and Fleurent (1991), we refer to a game to be scheduled (while not scheduled to any day), as a *free* game. A schedule where an NHL playable day has been assigned to every free game is referred to as a *complete schedule*. The basic problem of this thesis can then be stated as follows. Given the free games for an NHL regular season, the corresponding set D of playable dates provided by the NHL, and the set $D_t \subseteq D$ of arena dates for each NHL team t, the NHL scheduling problem in this thesis consists in scheduling every free game $a@h$ to a playable date $d \in D$, as to build a complete schedule having the minimum number of violations (if any) of the constraints from C_1 to C_9.

3.3.1 The NHL scheduling problem in the literature

The constraints we enumerated in the preceding subsection are essentially the same as those that have been considered in the academic literature on the NHL scheduling problem. In particular, only the constraints C_7, C_8 and C_9 are not taken into account in the mathematical model introduced by Fleurent (1987). That model deals with all the constraints from C_0 to C_6 as *hard* constraints, which cannot be violated, except for C_3, which is treated as a *soft* constraint and for which violations are allowed but penalized in the objective function. The general description of the NHL scheduling problem in Fleurent (1987), however, does

include the constraint C_7 and also a constraint enforcing a team to be scheduled to play at home on its first arena date after a trip of more than seven days. Those two constraints are, in fact, characterized as hard constraints in that description of the problem. Apart from the violations of C_3, the objective function of the mathematical model referred above incorporates, in a weighted sum, both the number of games to be played on weekdays (from Monday to Thursday) and the total distance to be travelled by all teams of the league. In this thesis, among the specified constraints, we denote the set of hard constraints considered in the model by Fleurent (1987) as

$$\mathsf{C}_F = \{\mathsf{C}_0, \mathsf{C}_1, \mathsf{C}_2, \mathsf{C}_4, \mathsf{C}_5, \mathsf{C}_6\}.$$

With regard to the work by Costa (1995), the constraints from C_0 to C_3 are treated as soft constraints, and the ones from C_4 to C_8 are considered hard constraints, except for C_5, which is not mentioned by the author. In particular, during the solution process, violations for the constraint C_0 are actually allowed only for the cases where either two away games or two home games are scheduled on the same day for a team. However, at the end of Costa's evolutionary approach, which holds a *population* of schedules in each iteration, any final solution violating C_0 is rejected.

As noticed in the preceding chapter, Craig *et al.* (2009) do not mention other constraints than C_0. However, they penalize any road trip made up of more than three games, and both C_7 and C_8 could then be interpreted as soft constraints in their work.

Table 3.2 *Constraint violations in official NHL schedules* For each constraint (*ctr*), number of violations in the actual NHL schedules of the last five seasons.

ctr	2009-10	2010-11	2011-12	2012-13	2013-14
C_1	0	0	0	0	0
C_2	0	0	0	0	0
C_3	3	3	0	0	0
C_4	12	14	9	10	10
C_5	6	16	19	3	8
C_6	87	66	75	88	33
C_7	1	0	1	1	1
C_8	1	0	1	2	0
C_9	0	0	0	0	0
Total	110	99	105	104	52

On the whole, Table 3.3 summarises the constraints that had (and that had not) been considered in the literature. Although none of those references has explicitly defined the constraint C_9, it has been motivated by our attempt to mimic a more realistic representation of consecutive road trips in NHL schedules. In fact, our definition of *trip* introduced in the preceding subsection is based on the approach for generating road trips that was used in Ferland and Fleurent (1991). Specifically, that approach does not allow a team to have byes made up of three or more consecutive days in the course of a single road trip. With regard to Costa (1995), this requirement was relaxed by one day, as he tries to avoid byes of four or more consecutive days during a road trip.

Table 3.3 *Constraints of the NHL scheduling problem in the literature* Comparison of the constraints (*ctr*) in this thesis ($\mathsf{C}_0, \mathsf{C}_1, \ldots, \mathsf{C}_9$) with those that are, or that are not ($\times$), taken into account in Craig *et al.* (2009) (CWB-2009), in Costa (1995) (C-1995), in Ferland and Fleurent (1991) (FF-1991), and in Fleurent (1987) (F-1987). Some of the constraints have been treated as *hard* constraints, and others as *soft* constraints. In particular, with regard to the (*soft**) constraint C_0 in C-1995, only violations where either two away games or two home games are scheduled to the same day for a team are allowed during the solution process; but as well as in CWB-2009, any final solution that violates C_0 is rejected.

ctr	F-1987	FF-1991	C-1995	CWB-2009
C_0	*hard*	*hard*	*soft**	*soft**
C_1	*hard*	*hard*	*soft*	$\times$
C_2	*hard*	*hard*	*soft*	$\times$
C_3	*soft*	*soft*	*soft*	$\times$
C_4	*hard*	*hard*	*hard*	$\times$
C_5	*hard*	*hard*	$\times$	$\times$
C_6	*hard*	*hard*	*hard*	$\times$
C_7	$\times$	$\times$	*hard*	*soft*
C_8	$\times$	$\times$	*hard*	*soft*
C_9	$\times$	$\times$	$\times$	$\times$

The surprisingly high number of violations for the constraints C_4 and C_6 in the most recent NHL schedules (see Table 3.2) does not support, nowadays, the assumption that these constraints should be considered as hard constraints with the same parameter values (900 miles in the description of C_4, and 14 and 30 days in the description of C_6) that were used by Ferland and Fleurent (1991) and by Costa (1995) in the 1990s.

In the next chapter, we first formulate the NHL scheduling problem as an integer linear programming and then, we present some computational results obtained from a commercial state-of-the-art solver on different configurations of hard and soft constraints.

CHAPTER 4

MATHEMATICAL MODELING

In this chapter, we propose the first integer linear programming model for the NHL scheduling problem which takes into account all its constraints considered in the literature, in addition to the minimization of total travel distance for the league. Some computational results obtained from a commercial state-of-the-art solver on several variants of the model are also reported.

Section 4.1 specifies the data of the problem, the parameters for the model, and the mathematical formulations that cover all the constraints enumerated in the previous chapter. The formulation for minimizing total travel distance will be presented in Section 4.2.

4.1 Basic formulation

In this section, we present a formulation for each constraint of the problem, which leads to a feasibility model with $O(n^2m)$ variables for n teams and m days.

4.1.1 Data of the problem

The data of the problem are the following:

— n: number of teams in the league;

— m: number of calendar days in the regular season period;

— $T = \{1, 2, \ldots, n\}$: set of teams in the league;

— $D = \{1, 2, \ldots, m\}$: set of calendar days in the regular season period;

— $D_t = \{d_1^t, d_2^t, \ldots, d_{m_t}^t\}$: set of arena-available days for each team $t \in T$;

— $G_{a,h}$: number of games (visits) for each match $a@h$, where $a, h \in T$; and

— $dist(t', t'')$: distance between the arenas of teams t' and t'', for all $t', t'' \in T$.

4.1.2 Parameters of the model

Based on the description of the constraints of the problem in the preceding chapter, the following are the parameters for the model:

— $\delta_{a,h}^{VV}$: minimum number of days between (re)visits for the match $a@h$, where $a, h \in T$;

— ξ^{DD}: maximum daily travel distance for a team;

— ξ^{AB}: maximum number of consecutive "away byes" (during a same trip);

— ξ^{AG}: maximum number of away games in any road trip for a team;

— ξ^{AD}: maximum duration, in number of days, of any road trip for a team;

— ξ^{WG}: minimum number of games per week for a team;

— δ^{RR}_{far}: minimum number of days between two consecutive road trips for a team that has to travel long distances (more than ξ^{DD}) both back home from the last game in the first trip and from home to the first game in the second trip;

— $\delta^{RR}_{\text{near}}$: minimum number of days between two road trips for a team; that has to travel short distances (no more than ξ^{DD}) both back home from the last game in the first trip and from home to the first game in the second trip.

In particular, ξ^{DD}, on the maximum daily distance, can be seen as a threshold beyond which a distance is considered "too far" for a team to travel in order to play back-to-back games.

4.1.3 Variables

For any two teams $a, h \in T$ and any day $d \in D$, let $x^A_{a,d,h}$ be a binary variable defined as

$$x^A_{a,d,h} = \begin{cases} 1 & \text{if team } a \text{ plays away on day } d \text{ at the arena of team } h, \\ 0 & \text{otherwise.} \end{cases}$$

In addition, for any team $t \in T$ and any day $d \in D$, let $x^H_{t,d}$, $x^B_{t,d}$, and $x^L_{t,d}$ be binary variables defined as

$$x^H_{t,d} = \begin{cases} 1 & \text{if team } t \text{ plays home on day } d, \\ 0 & \text{otherwise;} \end{cases}$$

$$x^B_{t,d} = \begin{cases} 1 & \text{if team } t \text{ does not play (has a bye) on day } d, \\ 0 & \text{otherwise;} \end{cases}$$

$$x^L_{t,d} = \begin{cases} 1 & \text{if team } t \text{ has at least } (\xi^{AB}+1) \text{ byes in a row, starting on day } d, \\ 0 & \text{otherwise.} \end{cases}$$

In the descriptions that follow, we might refer to $x^A_{a,d,h}$, $x^H_{t,d}$, $x^B_{t,d}$, and $x^L_{t,d}$ as *away*, *home*, *bye*, and *long-bye* variables, respectively.

4.1.4 Constraints

We now present the formulation of each constraint of the problem with a short informal description for each of them to (hopefully) make the model easier to understand.

Restricting a team to play at most one game per day On any day $d \in D$, exactly one of the following cases must happen for any team $t \in T$:

1) t plays an away game (at the arena of a single opponent);

2) t plays a home game;

3) t does not play any game.

In our model, this is represented by the following constraints:

$$\sum_{h \in T \setminus \{t\}} x^A_{t,d,h} + x^H_{t,d} + x^B_{t,d} = 1 \qquad \forall t \in T,\ \forall d \in D \qquad (4.1)$$

Linking host and visiting team If a team $t \in T$ plays home on a day $d \in D$ then some other team $a \in T$ must be visiting t on day d. This is represented by the following constraints:

$$\sum_{a \in T \setminus \{t\}} x^A_{a,d,t} - x^H_{t,d} = 0 \qquad \forall t \in T,\ \forall d \in D \qquad (4.2)$$

Fixing non-meaningful "self-visit" variables For any team $t \in T$ and any day $d \in D$, the variable $x^A_{t,d,t}$ is set to 0.

$$x^A_{t,d,t} = 0 \qquad \forall t \in T, \forall d \in D \qquad (4.3)$$

Linking variables $x^B_{t,d}$ and $x^L_{t,d}$ For any team $t \in T$ and any day $d \in D$, such that $d <= m - \xi^{AB}$, the variable $x^L_{t,d}$ should be set to 1 if and only if t does not play from day d to day $d + \xi^{AB}$, which can be written as the logical connective

$$x^L_{t,d} = 1 \iff x^B_{t,d} = x^B_{t,d+1} = x^B_{t,d+2} = \cdots = x^B_{t,d+\xi^{AB}} = 1.$$

The idea is that $x^L_{t,d} = 1$ indicates that day d is not within a road trip period for team t and, in particular, if t plays away on the previous day, $d - 1$, then it must return home before playing the next game (even when this is also an away game). In terms of linear inequalities,

we add then the following constraints to the model:

$$x^L_{t,d} \leq x^B_{t,d} \qquad \forall t \in T,\, \forall d \in \{1,2,\ldots,m-\xi^{AB}\} \tag{4.4}$$
$$x^L_{t,d} \leq x^B_{t,d+1} \qquad \forall t \in T,\, \forall d \in \{1,2,\ldots,m-\xi^{AB}\} \tag{4.5}$$
$$x^L_{t,d} \leq x^B_{t,d+2} \qquad \forall t \in T,\, \forall d \in \{1,2,\ldots,m-\xi^{AB}\} \tag{4.6}$$
$$\vdots$$
$$x^L_{t,d} \leq x^B_{t,d+\xi^{AB}} \qquad \forall t \in T,\, \forall d \in \{1,2,\ldots,m-\xi^{AB}\} \tag{4.7}$$
$$x^L_{t,d} \geq \sum_{k=0}^{\xi^{AB}} x^B_{t,d+k} - \xi^{AB} \qquad \forall t \in T,\, \forall d \in \{1,2,\ldots,m-\xi^{AB}\} \tag{4.8}$$

Setting the number of games for each pair of teams For each two teams $a \in T$ and $h \in T$, exactly a given number $G_{a,h}$ of games must be scheduled for the match a@ h:

$$\sum_{d \in D} x^A_{a,d,h} = G_{a,h} \qquad \forall a \in T,\, \forall h \in T \tag{4.9}$$

Constraint $\mathbf{C}_1$ (*arena availability for home games*) Every home variable for a non arena-available day of a team is fixed to value 0:

$$x^H_{t,d} = 0 \qquad \forall t \in T,\, \forall d \in D \setminus D_t \tag{4.10}$$

Constraint $\mathbf{C}_2$ (*maximum number of games over three days*) Every sequence of three days for a team is forced to have at least one bye:

$$\sum_{k=0}^{2} x^B_{t,d+k} \geq 1 \qquad \forall t \in T,\, \forall d \in D,\, d \leq m-2 \tag{4.11}$$

Constraint $\mathbf{C}_3$ (*maximum number of games over five days*) Every sequence of five days for a team is forced to have at least two byes:

$$\sum_{k=0}^{4} x^B_{t,d+k} \geq 2 \qquad \forall t \in T,\, \forall d \in D,\, d \leq m-4 \tag{4.12}$$

Constraint C_4 (*maximum daily travel distance*) For any two consecutive days, at least one of every two variables corresponding to two distant arenas is forced to take value 0:

$$x^A_{t,d,h'} + x^A_{t,d+1,h''} \leq 1 \qquad \forall t \in T,\, \forall d \in D \setminus \{m\},\, \forall h', h'' \in T, dist(h', h'') > \xi^{DD} \tag{4.13}$$

$$x^H_{t,d} + x^A_{t,d+1,h'} \leq 1 \qquad \forall t \in T,\, \forall d \in D \setminus \{m\},\, \forall h' \in T,\, dist(t, h') > \xi^{DD} \tag{4.14}$$

$$x^A_{t,d,h'} + x^H_{t,d+1} \leq 1 \qquad \forall t \in T,\, \forall d \in D \setminus \{m\},\, \forall h' \in T,\, dist(h', t) > \xi^{DD} \tag{4.15}$$

Constraint C_5 (*minimum number of games in a week*) Regarding every week of the season, say w, at least ξ^{WG} bye variables corresponding to the days, $d^1_w, d^2_w, \ldots, d^7_w$, of that week are forced to take value 0:

$$x^B_{t,d^1_w} + x^B_{t,d^2_w} + \ldots + x^B_{t,d^7_w} \leq 7 - \xi^{WG} \qquad \forall t \in T,\, \forall w \in \{1, 2, \ldots, \lfloor n/7 \rfloor\} \tag{4.16}$$

Constraint C_6 (*minimum number of days between revisits*) Regarding every match $a@h$, for each interval of $1 + \delta^{VV}_{a,h}$ days, no more than one away variable is allowed to take value 1:

$$\sum_{k=0}^{\delta^{VV}_{a,h}} x^A_{a,d+k,h} \leq 1 \qquad \forall a, h \in T,\, \forall d \in D, d \leq m - \delta^{VV}_{a,h} \tag{4.17}$$

Constraint C_7 (*maximum number of games in a trip*) The idea behind the formulation of this constraint is the following. Consider at first the definition of road trip from the previous chapter (p. 34), which allows no more than two byes between two consecutive games during the same trip (i.e., $\xi^{AB} = 2$), and consider also that a trip must have no more than seven games (i.e., $\xi^{AG} = 7$). In this case, any trip that violates constraint C_7 has at least eight games. And any trip with eight games lasts between eight days (when there is one game for each day) and 22 days (when there is one game for each three, $\xi^{AB} + 1$, consecutive days). Therefore, to prevent trips from having eight or more games, every sequence of ℓ days, with $\ell \in \{8, 9, \ldots, 22\}$, must have at least $\ell - 7$ byes or at least one "trip breaker" (a home game or a bye lasting more than two days), which can be expressed by the following linear inequalities:

$$\sum_{k=0}^{\ell-1} \left(x^B_{t,d+k} + (\ell - 7) x^H_{t,d+k} + (\ell - 8) x^L_{t,d+k} \right) \geq \ell - 7 \qquad \forall t \in T,\, \forall d \in D, d \leq m - \ell.$$

For the general case, given the parameters ξ^{AG} and ξ^{AB}, every sequence of ℓ days for a team, with $\ell \in \{1 + \xi^{AG}, \ldots, 1 + \xi^{AG} \cdot (\xi^{AB} + 1)\}$, must have at least $\ell - \xi^{AG}$ byes or at least

one "trip breaker" (a home game or a bye longer than ξ^{AB} days):

$$\sum_{k=0}^{\ell-1} \left(x^B_{t,d+k} + (\ell - \xi^{AG})x^H_{t,d+k} + (\ell - \xi^{AG} - 1)x^L_{t,d+k}\right) \geq \ell - \xi^{AG} \qquad (4.18)$$

$$\forall t \in T,\, \forall d \in D, d \leq m - \ell.$$

Constraint C_8 (*maximum number of days in a trip*) The idea on the formulation of this constraint is similar to the previous one, on C_7, as we look at the shortest sequences of days that would potentially violates the constraint C_8. At first, we describe the formulation for the particular case from the previous chapter (p. 34), where a trip should last no more than 14 days ($\xi^{AD} = 14$) and away byes should last no more than two days ($\xi^{AB} = 2$). Regarding any given team, we impose here that if a team t has an away game on day d (i.e., $x^H_{t,d} + x^B_{t,d} = 0$) and another away game on day $d + 14$ (i.e., $x^H_{t,d+14} + x^B_{t,d+14} = 0$) then at least one "trip breaker" (a home game or a bye lasting more than two days) must exist between those two away games. Similar restriction is imposed on days d and $d + 15$, and also on days d and $d + 16$. We point out that, because two games in the same trip are never separated by more than two days, such restrictions on periods of 15, 16, and 17 days are enough to cover any trip that would potentially last more than 14 days (and then violates C_8). Hence, if a team has two away games separated by $\ell - 2$ days, with $\ell \in \{15, 16, 17\}$, the following linear inequalities must hold:

$$x^H_{t,d} + x^B_{t,d} + \sum_{k=1}^{\ell-2} \left(x^H_{t,d+k} + x^L_{t,d+k}\right) + x^H_{t,d+\ell-1} + x^B_{t,d+\ell-1} \geq 1$$

$$\forall t \in T,\, \forall d \in D, d \leq m - \ell.$$

For the general case, given the parameters ξ^{AD} and ξ^{AB}, if a team has away games on days d and $d+\ell-1$, with $\ell \in \{\xi^{AD}+1, \ldots, \xi^{AD}+\xi^{AB}+1\}$, then at least one "trip breaker" (a non-zero home variable or a non-zero long-bye variable) must exist between those two away games:

$$x^H_{t,d} + x^B_{t,d} + \sum_{k=1}^{\ell-2} \left(x^H_{t,d+k} + x^L_{t,d+k}\right) + x^H_{t,d+\ell-1} + x^B_{t,d+\ell-1} \geq 1 \qquad (4.19)$$

$$\forall t \in T,\, \forall d \in D, d \leq m - \ell.$$

Constraint C_9 (*minimum number of days between two trips*) The basic idea on the formulation of this constraint is to look at each short period of days that would potentially violate constraint C_9 for a team and specify some inequalities preventing such violations from

actually happening. In particular, given a traveling team t and the set F_t of all its opponents located far from its own home arena, we look at each period of less than δ^{RR}_{far} days, say from day d to day $d+q$, with $q \in \{1,\ldots,\delta^{RR}_{\text{far}}-1\}$, and whenever t has no away games but one "trip breaker" (a home game or a long bye) during those q days, we impose that t is not scheduled to play away on both day $d-1$ and day $d+q+1$ against its opponents in F_t. Obviously, the idea is similar when considering only the opponents located close to the arena of the traveling team under consideration.

As an example, considerer the description of C_9 in the previous chapter (p. 36), where we have $\delta^{RR}_{\text{far}} = 3$ and $\delta^{RR}_{\text{near}} = 2$. Because away byes last no more than two days ($\xi^{AB} = 2$), the formulation of C_9 comes down to only the following two cases.

— Preventing a team of having away games on days d and $d+2$ when it has a home game on day $d+1$:

$$\sum_{h\in T\setminus\{t\}} x^A_{t,d,h} + x^H_{t,d+1} + \sum_{h\in T\setminus\{t\}} x^A_{t,d+2,h} \leq 2,$$

for each $t \in T$ and each $d \in D$ with $d < m-2$.

— Preventing a team of having away games on days d and $d+3$ when it has, in total, at least one home game on days $d+1$ and $d+2$:

$$2\cdot\sum_{h\in F_t} x^A_{t,d,h} + x^H_{t,d+1} + x^H_{t,d+2} + 2\cdot\sum_{h\in F_t} x^A_{t,d+3,h} \leq 4,$$

for each $t \in T$ and each $d \in D$ with $d < m-3$, where $F_t = \{h \in T : dist(h,t) > 900\}$.

For the general case, regarding a traveling team $t \in T$, any two away games $t@h'$ and $t@h''$ that are respectively the last game and the first game of two consecutive road trips for t must be scheduled at least δ^{RR}_{far} days apart from each other if both the distances between h' and t, and between t and h'', are *far* (more than ξ^{DD} miles), or at least $\delta^{RR}_{\text{near}}$ days apart from each other, otherwise. In particular, if the arenas of both h' and h'' are *far* from the arena of t, we impose then the following constraints:

$$q\cdot\sum_{h\in F_t} x^A_{t,d,h} + \sum_{k=1}^{q}\left(x^H_{t,d+k} + x^L_{t,d+k} - (q-1)\cdot\sum_{h\in T} x^A_{t,d+k,h}\right) + q\cdot\sum_{h\in F_t} x^A_{t,d+q+1,h} \leq 2q \tag{4.20}$$

$$\forall t \in T,\, q \in \{1,\ldots,\delta^{RR}_{\text{far}}-1\}, \forall d \in D, d < m-q,$$

$$F_t = \{h \in T : dist(h,t) > \xi^{DD}\}.$$

Similarly, if the arena of either h' or h'' is *close* to the arena of t, we impose then the following constraints:

$$q \cdot \sum_{h \in F_t} x^A_{t,d,h} + \sum_{k=1}^{q} \left(x^H_{t,d+k} + x^L_{t,d+k} - (q-1) \cdot \sum_{h \in T} x^A_{t,d+k,h} \right) + q \cdot \sum_{h \in T \setminus F_t} x^A_{t,d+q+1,h} \leq 2q \tag{4.21}$$

and

$$q \cdot \sum_{h \in T \setminus F_t} x^A_{t,d,h} + \sum_{k=1}^{q} \left(x^H_{t,d+k} + x^L_{t,d+k} - (q-1) \cdot \sum_{h \in T} x^A_{t,d+k,h} \right) + q \cdot \sum_{h \in F_t} x^A_{t,d+q+1,h} \leq 2q \tag{4.22}$$

$$\forall t \in T,\, q \in \{1, \ldots, \delta^{RR}_{\text{near}} - 1\}, \forall d \in D, d < m - q,$$

$$F_t = \{h \in T : \, dist(h,t) > \xi^{DD}\}.$$

4.2 Formulation for minimizing total travel distance

In this section, we present a formulation that involves the minimization of total travel distance, which, compared to the basic formulation from the preceding section, results in a much higher number of variables and constraints. In particular, the resulting model has $O(n^3 m)$ variables for n teams and m days, which is in line with other sports scheduling formulations in the literature (Trick, 2005; Ribeiro, 2012).

The formulation introduced in the preceding section is essentially based on the *place* that every team must be on each day of the season. In order to solve the NHL scheduling problem with minimization of total travel distance, we add then to the basic formulation new variables that, regarding every team, account for the sequence of *moves* from one place to another.

In fact, the new variables can be described, for each $\ell \in \{1, 2, \ldots, \xi^{AB} + 1\}$, as follows:

$$y^{\ell}_{t,d,h',h''} = \begin{cases} 1 & \text{if team } t \text{ is at the home location of team } h' \text{ on day } d \\ & \text{and moves to the home location of team } h'' \text{ on day } d + \ell, \\ 0 & \text{otherwise,} \end{cases}$$

for any teams $t, h', h'' \in T$, and any day $d \in D$ such that $d \leq m - \ell$. In particular, indices h' and h'' can both refer to team t itself.

Generally speaking, for any $\ell \in \{1, 2, \ldots, \xi^{AB}\}$, having $y^{\ell}_{t,d,h',h''} = 1$ indicates that t plays at the arena of h' on day d and then plays at the arena of h'' on day $d + \ell$. However, in

addition to that case, if $\ell = \xi^{AB} + 1$ then having $y^{\ell}_{t,d,h',h''} = 1$ might also indicate that either t plays at the arena of h' on day d and then has a forced home bye on day $d+\ell$ (team t must return home due a long-bye period), or t has a forced home bye on day d and then plays at h'' on day $d+\ell$ (team t leaves home after a long-bye period).

These new variables are linked to those in the basic formulation by mean of the following constraints, which are logically grouped for easy understanding.

— two consecutive away games:

$$y^{\ell}_{t,d,h',h''} \geq x^{A}_{t,d,h'} + \sum_{k=1}^{\ell-1} x^{B}_{t,d+k} + x^{A}_{t,d+\ell,h''} - \ell \tag{4.23}$$
$$\forall t, h', h'' \in T,\ t \neq h',\ t \neq h'',\ \ell \in \{1, \ldots, \xi^{AB} + 1\}, \forall d \in D, d \leq m - \ell;$$

— home game and then away game:

$$y^{\ell}_{t,d,t,h'} \geq x^{H}_{t,d} + \sum_{k=1}^{\ell-1} x^{B}_{t,d+k} + x^{A}_{t,d+\ell,h'} - \ell \tag{4.24}$$
$$\forall t, h' \in T,\ t \neq h',\ \ell \in \{1, \ldots, \xi^{AB} + 1\}, \forall d \in D, d \leq m - \ell;$$

— away game and then home game:

$$y^{\ell}_{t,d,h',t} \geq x^{A}_{t,d,h'} + \sum_{k=1}^{\ell-1} x^{B}_{t,d+k} + x^{H}_{t,d+\ell} - \ell \tag{4.25}$$
$$\forall t, h' \in T,\ t \neq h',\ \ell \in \{1, \ldots, \xi^{AB} + 1\}, \forall d \in D, d \leq m - \ell;$$

— away game and then long-bye period (forced home bye):

$$y^{\ell}_{t,d,h',t} \geq x^{A}_{t,d,h'} + \sum_{k=1}^{\ell-1} x^{B}_{t,d+k} + x^{B}_{t,d+\ell} - \ell \tag{4.26}$$
$$\forall t, h', \in T,\ t \neq h',\ \ell = \xi^{AB} + 1, \forall d \in D, d \leq m - \ell;$$

— long-bye period (forced home bye) and then away game:

$$y^{\ell}_{t,d,t,h'} \geq x^{B}_{t,d} + \sum_{k=1}^{\ell-1} x^{B}_{t,d+k} + x^{A}_{t,d+\ell,h'} - \ell \tag{4.27}$$
$$\forall t, h', \in T,\ t \neq h',\ \ell = \xi^{AB} + 1, \forall d \in D, d \leq m - \ell;$$

— first game (away) within the first $\xi^{AB}+1$ days of the season:

$$y^{\ell}_{t,\ell,t,h'} \geq \sum_{d=1}^{\ell-1} x^{B}_{t,d} + x^{A}_{t,\ell,h'} - \ell + 1 \qquad (4.28)$$
$$\forall t, h', \in T,\ t \neq h',\ \ell \in \{1,\ldots,\xi^{AB}+1\}; and$$

— last game (away) within the last $\xi^{AB}+1$ days of the season:

$$y^{\ell}_{t,m-\ell+1,h',t} \geq x^{A}_{t,m-\ell+1,h'} + \sum_{d=m-\ell+2}^{m} x^{B}_{t,d} - \ell + 1 \qquad (4.29)$$
$$\forall t, h', \in T,\ t \neq h',\ \ell \in \{1,\ldots,\xi^{AB}+1\}.$$

Hence, in particular, the trip that a team travels from home for an away game in the first $\xi^{AB}+1$ days of the season is taken into account in (4.28), and the trip that a team travels for an away game in the last $\xi^{AB}+1$ days of the season to home is taken into account in (4.29).

Finally, with the y variables, minimizing the total travel distance comes down to the following straightforward linear objective function:

$$minimize \quad \sum_{\ell=1}^{\xi^{AB}+1} \sum_{t=1}^{n} \sum_{d=1}^{m-\ell} \sum_{h'=1}^{n} \sum_{h''}^{n} dist(h', h'') \cdot y^{\ell}_{t,d,h',h''}. \qquad (4.30)$$

4.3 Computational experiments

In this section, we report several results from an implementation of the preceding mathematical model in a commercial state-of-the-art solver, namely the IBM ILOG CPLEX® 12.6.

In fact, all the models resulting from the relaxation of any of the constraints described in the preceding section have been implemented in C++ and use the Concert Technology C++ API of CPLEX®. Here, by "relaxation" we mean either the omission of constraints or the addition of artificial variables to the constraints and corresponding penalty in the objective function. It was run under Linux on a 3.07GHz Intel(R) Xeon(R) X5675 processor (only one processor was used) with 94.5G of RAM. For each experiment, the maximum time which CPLEX® was allowed to run had been set (through the parameter `IloCplex::TiLim`) to 48 hours, and the number of threads had also been limit to a single one (through the parameter `IloCplex::Threads`). Test were performed on different values for the MIP emphasis parameter (`MIPEmphasis`), but because it had no noticeable impact in the quality of the overall results, we limit this reporting to those obtained with this parameter set as *balanced*, which orients the search for a solution toward both optimality and integer feasibility.

4.3.1 Models and parameters

The results reported in this chapter are limited to four models and two settings of parameters. Furthermore, every combination model-and-parameters was tested both with and without minimization of total travel distance. We use the term *MinDist* to denote the presence of minimization of distance in the model at hand. Each model refers to only a particular subset of constraints being relaxed, but always penalized in the objective function. And each setting of parameters fix different values for the revisit-gap parameters in the models.

To be more precise, the models we refer to are characterized by the following description, which is summarized in Table 4.1.

Model I This model relaxes only the constraint C_5 on the minimum number of games to be played by a team during every week of the season.

Model II In addition to C_5, this model relaxes the constraints C_7, C_8, C_9 on the maximum number of games per trip, on the maximum duration for a trip, and on the minimum number of days between two consecutive trips for a team.

Model III In addition to the previous relaxations, on C_5, C_7, C_8, and C_9, this model relaxes the constraints C_4 and C_6, on the daily travel distance and on the minimum number of days between two games involving the same pair of teams in the same place (the revisit gap).

Model IV This model relaxes all the nine constraints, from C_1 to C_9 in our descriptions of the problem.

Table 4.1 *Models for the computational experiments with CPLEX®* Characterization of the models with regard to each constraint of the problem.

Constraint	Model I	Model II	Model III	Model IV
C_1 : `arenaAvailability`	*hard*	*hard*	*hard*	*soft*
C_2 : `max2GamesOver3Days`	*hard*	*hard*	*hard*	*soft*
C_3 : `max3GamesOver5Days`	*hard*	*hard*	*hard*	*soft*
C_4 : `maxDistancePerDay`	*hard*	*hard*	*soft*	*soft*
C_5 : `minNGamesPerWeek`	*soft*	*soft*	*soft*	*soft*
C_6 : `minNDaysBetweenRevisits`	*hard*	*hard*	*soft*	*soft*
C_7 : `maxNGamesPerTrip`	*hard*	*soft*	*soft*	*soft*
C_8 : `maxNDaysPerTrip`	*hard*	*soft*	*soft*	*soft*
C_9 : `minNDaysBetweenTrips`	*hard*	*soft*	*soft*	*soft*

In all these models, the objective function is to be minimized, as it is defined by a weighted

summation of variables representing the violation of the relaxed constraints, in addition to the total distance (when *MinDist* is considered in the model). For the computational experiments, we have set the weight (penalty parameter) w_i for the violation of a constraint C_i to the following values: $w_1 = 10\,000$, $w_2 = 1\,000$, $w_3 = 100$, $w_5 = 1$, and $w_i = 10$ for each constraint C_i with $i \in \{4, 6, 7, 8, 9\}$.

As for the parameters, we report results for two settings, *LGap* and *SGap*, which differ only on the values for the revisit gaps. Indeed, both settings use the following values: $\xi^{DD} = 900$, $\xi^{AB} = 2$, $\xi^{AG} = 7$, $\xi^{AD} = 14$, $\xi^{WG} = 2$, $\delta^{RR}_{\text{far}} = 3$, and $\delta^{RR}_{\text{near}} = 2$; and they differ on the revisit gaps ($\delta^{VV}_{a,h}$) as follows:

Long revisit gaps (*LGap*) The minimum number of days between two games (revisits) for each match is set to 13 and to 29 for intra-division and for interdivisional revisits, respectively, as in Fleurent (1987).

Short revisit gaps (*SGap*) The minimum number of days between two games (revisits) for each match is set to only 5 and to only 11 for intra-division and for interdivisional revisits, respectively.

4.3.2 Problem instances

The data for the NHL scheduling problem are made up by specifying *1)* the arrangement of the League (comprising the distance between each pair of arenas), *2)* the structure of the regular season (with all the games that must be scheduled), and *3)* the arena dates for each team. As we have already stated, changes in the structures of both the League and its regular season usually occur (when they do) on a very small scale from one year to the next. On the other hand, the availability of arenas is expected to be quite irregular over different seasons. To our knowledge, no such data are openly available, even for the past years. We have decided, therefore, to create our own test problems, which we categorize into the three following classes.

Entire-league instances Instances in this class were generated by keeping all home dates in an actual NHL schedule and adding to it some *extra arena-available dates*. Such dates were randomly chosen from the corresponding regular-season period. We denote these instances by `NHL`α`-`$\beta\gamma$, where α refers to the starting year of the concerning season; β is the number of extra arena dates per team; and γ is a letter to identify different instances for the same year (α) and number of extra dates (β). For example, `NHL13-4a` stands for a particular instance identified by `a` and having 4 additional home dates per team with regard to the NHL schedule of the 2013-14 regular season. Given any official NHL schedule, the only two parameters in our instance generator for this class are the

Table 4.2 *Instances for the computational experiments* Information about some of the problem instances used in this thesis: numbers of conferences (*#confs*), divisions (*#divs*), teams (*#teams*), games in a complete schedule (*#games*), games per team (*#tgames*), home games per team (*#hgames*), and extra arena-available dates per team (*#edays*).

Instance	*#confs*	*#divs*	*#teams*	*#games*	*#tgames*	*#hgames*	*#edays*
Entire-league instances							
NHL09-00a	2	6	30	1230	82	41	0
NHL10-00a	2	6	30	1230	82	41	0
NHL11-00a	2	6	30	1230	82	41	0
NHL13-00a	2	4	30	1230	82	41	0
NHL09-04a	2	6	30	1230	82	41	4
NHL10-04a	2	6	30	1230	82	41	4
NHL11-04a	2	6	30	1230	82	41	4
NHL13-04a	2	4	30	1230	82	41	4
NHL09-08a	2	6	30	1230	82	41	8
NHL10-08a	2	6	30	1230	82	41	8
NHL11-08a	2	6	30	1230	82	41	8
NHL13-08a	2	4	30	1230	82	41	8
Partial-league instances							
NHL11-1208-76	1	2	8	304	76	38	3
NHL13-1208-76	1	2	8	304	76	38	3
NHL11-1208-74	1	2	8	296	74	37	4
NHL13-1208-74	1	2	8	296	74	37	4
NHL11-1208-72	1	2	8	288	72	36	5
NHL11-1210-74	1	2	10	370	74	37	4
NHL13-1212-76	1	2	12	456	76	38	3
NHL11-1312-78	1	3	12	468	78	39	2
NHL11-1312-72	1	3	12	432	72	36	5
NHL13-1216-74	1	2	16	592	74	37	4
NHL13-2416-78	2	4	16	624	78	39	2
NHL13-2416-74	2	4	16	592	74	37	4
NHL11-2624-74	2	6	24	888	74	37	4
The 2012 instances							
NHL12-1315a	1	3	15	360	48	24	0
NHL12-1315b	1	3	15	360	48	24	0

number and the "type" of extra dates per team. The *type* parameter indicates the dates within the regular-season period that are the candidates to become extra dates for the instance being generated. In fact, this parameter specifies which of the following two

strategies is applied: *(1)* the extra dates are chosen, for any team t, only among those days when t had not been scheduled to play (neither home nor away) in the actual NHL schedule; or *(2)* the extra dates are selected among any non-home dates (even those when t had been scheduled to play away) in the actual NHL schedule. This is based upon the rationale that finding a solution for the instances in which all the new dates, regarding the actual NHL schedule, are indeed extra dates may be easier than scheduling games for instances in which some NHL away dates had also been selected as "extra" dates. Evidences for such premise is to be gathered throughout the experiments reported in this thesis. We refer to those two types of dates as *true-extra* dates and *pseudo-extra* dates, respectively.

Partial-league instances Instances in this class are formed by only some randomly chosen subset of the teams from the League. The home dates in the corresponding actual NHL schedule for the chosen teams are taken as arena-available dates in these instances. But the numbers of games to be scheduled are changed. They are proportionally increased for each pair of teams until every team would have to play in this "partial league" almost the same number of games it plays in the actual schedule. Obviously, even with such small artificial leagues, the idea is to mimic the unavailability of arena dates from the real situation, where a limited number of dates are reported to be available for the home games of many teams.

The 2012 instances There are only two instances in this class, named `NHL12-1315a` (for the Eastern Conference schedule) and `NHL12-1315b` (for the Western Conference schedule), each one made up from the 15 teams (and their home dates) on the NHL 2012-13 season for a particular conference. The actual schedule for that season, during which only 720 games were played, is in fact formed by two independent schedules, one for each conference. In particular, no game was played between a team from the Eastern Conference and another from the Western Conference during that shortened season due to a labour dispute that started in 2012 and lasted for a few months.

Table 4.2 outline some information about the instances of these three classes that we use for the computational experiments reported in this chapter.

We must remark that some instances also provide a few pre-scheduled games if the corresponding actual NHL schedule includes some special games that are not supposed to be rescheduled. Examples of such games are Première games (which are played in Europe), Winter Classic games (which are usually fixed in the middle of the season), Stadium Series games, and Heritage Classic games. Furthermore, only days that in the corresponding actual schedule have at least one scheduled game are considered as "playable days" in our instances. In particular, our implementations does not schedule games to special days that involves, for

example, Christmas holidays, All Star game week, and Olympic break. In addition, for a team that, in the beginning of the season, plays a Première game, no game is scheduled in a period of at least four days after the last of those games (in Europe).

The results are grouped by class of instances throughout the following subsections. We start reporting on the partial-league instances, which are expected to be somewhat less difficult to solve than the entire-league instances.

4.3.3 Results for the *partial-league* instances

Results for the partial-league instances are presented in tables 4.3 and 4.4 respectively for the cases with and without minimization of total travel distance. These tables show the running times and output solution status from CPLEX® on the four models (different relaxed constraints) with the two revisit-gap configurations we mentioned earlier. For a model in which a particular constraint has been relaxed, the reference to the constraint, C_i, is enclosed between brackets, $[\mathsf{C}_i]$, on the top of the tables. The results are separated into two groups: one for each configuration of the revisit-gap parameters, which are either long (*LGap*) or short (*SGap*). The solution status (*Sol*) indicates whether (*feas*) or not (–) a feasible solution, not necessarily the optimal (**opt**), has been found within the time limit of 48 hours.

An inspection of the solution statuses in Table 4.3 indicates that, for large revisit gaps, 73% of the instances have been solved either to proved optimality (42%) or at least to feasibility (31%). That proportion is much larger for short revisit gaps (96%), where almost all instances were solved to optimality. Even though the short revisit gaps were intended to facilitate the search for feasible solutions with regard to the C_6 constraints, the results on short revisit gaps are quite remarkable, particularly for instances with 16 and 24 teams, which have no more than four extra arena dates per team.

As for the experiments involving the minimization of total travel distance (*MinDist*), although slightly more instances are solved to feasibility for the small than for the large revisit-gap configuration, the results in Table 4.4 show that the solver is somewhat disappointing in this case. In fact, by a deeper inspection of the feasible solutions from this part of the experiments and by comparing them with the corresponding feasible solutions referred in Table 4.3, we noticed that even though the total distances have indeed decreased for most of the cases, in general it happened to the price of higher number of constraint violations. Exception to this disappointing effect did occur with Model I, where only the C_5 constraints are allowed to be violated. Indeed, the total distance on every feasible solution in Table 4.4 for Model I with parameters *SGap+MinDist* happens to be between 4 and 16% lower than the distance on the corresponding (optimal) solutions without minimization of distance from Table 4.3. And this, at the price of no more than three additional violations of C_5 in the

same comparison.

Now, comparing the overall results in these two tables indicates that the models involving *MinDist* are indeed much more difficult to be solved than their corresponding configurations without considering the total distance. In fact, solutions (not necessarily optimal) were found only for 38% of the instances when the models incorporate *MinDist*, as opposed to more than 84% for the other case. This is somewhat in agreement with what we expected, because, in particular, the models with *MinDist* involves a huge number of additional constraints and variables.

In both tables, the results also suggest that shorter values for the revisit-gap parameters make the problems easier for CPLEX® to solve them. In practice, when sequentially scheduling the games, once a game for a particular match is scheduled, imposing larger revisit gaps reduces the range of remaining feasible days for other games of the same match. In a MIP solver, however, it could be expected that such reduction in the number of possible assignments would speed up the search for a solution, which does not seem to occur in the case of CPLEX® for any of our experiments in this chapter.

We also notice that, when we compare the results on the different models, these experiments support the general idea that solving such problems becomes less difficult when more (hard) constraints are present in the model, as in general it reduces the solution space of the problem.

4.3.4 Results for the *2012* instances

The 2012 instances are especially interesting for the application of our models in this chapter, as they are entirely based in the official NHL schedule while having numbers of games and teams that, considering the results in the preceding subsection, can probably be dealt with by our C++ CPLEX® implementations. Here, we also test the models I to IV, each one with the two configurations of revisit gaps, *LGap* and *SGap*, with and without minimization of total distance (*MinDist*). The corresponding running times and output solution status from CPLEX® are outlined in Table 4.5.

A first look at the results in this table reveals that the Western Conference instance, `NHL12-1315b`, seems surprisingly more difficult to solve than the other one, `NHL12-1315a`, which was solved to proved optimality on all models without *MinDist*, and at least to feasibility when total distance is to be minimized. In particular, regarding the models I and II, a feasible (optimal) solution was only found for the short revisit-gap configuration on Model II during 48 hours of search.

To further explore these results, we now focus in the instance `NHL12-1315a` for the particular case of Model III. Table 4.6 shows the constraint violations for the corresponding found

solutions, in addition to the increase in the total travel distance regarding the actual NHL schedule. Here, we notice that no constraint is violated by the optimal solutions obtained both for the easier case of small revisit gap and for the more realistic long revisit gap. But when minimization of total distance had been incorporated in those cases, some violations did occur in the best-found solutions. As for the value of the total distance, the particular case of the large revisit gap configuration also resulted in a worse solution than in the corresponding case without *MinDist*. This might suggest that, in general, the bigger models involving *MinDist* would need much more than 48 hours to possibly obtain reasonable solutions with respect to the total travel distance.

4.3.5 Results for the *entire-league* instances

Different from some rather promising results obtained for the previous classes of instances, CPLEX® failed to solve (in 48 hours) almost all the instances of this more realistic class. In fact, as shown in Table 4.7, CPLEX® was only able to solve five instances of the entire-league class through the most constrained models, I and II, for the "easier" short revisit-gap configuration without minimization of distance.

Curiously, all but one of the instances solved have no extra arena dates, which seems to suggest that, as expected, the search has been favored by the smaller search space of these cases.

Table 4.3 *Results for the partial-league instances without minimization of total travel distance* Running times and output solution status from CPLEX® on four different models. For a model in which a particular constraint has been relaxed, the reference to the constraint, C_i, is enclosed between brackets, $[\mathsf{C}_i]$, on the top of the table. The results are separated into two groups: one for each configuration of the revisit-gap parameters, which are either long (*LGap*) or short (*SGap*). The solution status (*Sol*) indicates whether (*feas*) or not (–) a feasible solution, not necessarily the optimal (**opt**), has been found within the time limit of 48 hours.

	Model I			Model II			Model III			Model IV		
	C_1	C_2	C_3	C_1	C_2	C_3	C_1	C_2	C_3	$[\mathsf{C}_1]$	$[\mathsf{C}_2]$	$[\mathsf{C}_3]$
	C_4	$[\mathsf{C}_5]$	C_6	C_4	$[\mathsf{C}_5]$	C_6	$[\mathsf{C}_4]$	$[\mathsf{C}_5]$	$[\mathsf{C}_6]$	$[\mathsf{C}_4]$	$[\mathsf{C}_5]$	$[\mathsf{C}_6]$
	C_7	C_8	C_9	$[\mathsf{C}_7]$	$[\mathsf{C}_8]$	$[\mathsf{C}_9]$	$[\mathsf{C}_7]$	$[\mathsf{C}_8]$	$[\mathsf{C}_9]$	$[\mathsf{C}_7]$	$[\mathsf{C}_8]$	$[\mathsf{C}_9]$
Param. *LGap*												
Instance	Time		*Sol*	Time		*Sol*	Time		*Sol*	Time		*Sol*
NHL11-1208-76	–	–		–	–		–		*feas*	–		*feas*
NHL13-1208-76	–	–		–	–		–		*feas*	–		*feas*
NHL11-1208-74	18h41m14s		**opt**	11m5s		**opt**	46h42m25s		**opt**	–		*feas*
NHL13-1208-74	15h6m0s		**opt**	–		*feas*	–		*feas*	–		*feas*
NHL11-1208-72	–	–		–	–		–		*feas*	–		*feas*
NHL11-1210-74	30m16s		**opt**	19m54s		**opt**	4h45m14s		**opt**	–		*feas*
NHL13-1212-76	15h13m55s		**opt**	1h22m46s		**opt**	–		*feas*	6h58m9s		**opt**
NHL11-1312-78	45h41m40s		**opt**	14h48m50s		**opt**	–		*feas*	–	–	
NHL11-1312-72	46h6m36s		**opt**	1h51m32s		**opt**	44h21m34s		**opt**	18h52m23s		**opt**
NHL13-1216-74	2h5m58s		**opt**	4h26m53s		**opt**	9h26m46s		**opt**	47h55m22s		**opt**
NHL13-2416-78	–	–		–	–		–		*feas*	–	–	
NHL13-2416-74	–	–		–		*feas*	–		*feas*	–	–	
NHL11-2624-74	11h49m29s		**opt**	41h8m32s		**opt**	–	–		–	–	
Param. *SGap*												
Instance	Time		*Sol*	Time		*Sol*	Time		*Sol*	Time		*Sol*
NHL11-1208-76	1m50s		**opt**	2m13s		**opt**	8m22s		**opt**	13m46s		**opt**
NHL13-1208-76	3m17s		**opt**	5m44s		**opt**	40m45s		**opt**	1h35m13s		**opt**
NHL11-1208-74	1m5s		**opt**	2m58s		**opt**	7m25s		**opt**	33m52s		**opt**
NHL13-1208-74	1m20s		**opt**	5m46s		**opt**	13m24s		**opt**	54m25s		**opt**
NHL11-1208-72	1m42s		**opt**	4m44s		**opt**	1h2m55s		**opt**	3h47m9s		**opt**
NHL11-1210-74	7m20s		**opt**	8m15s		**opt**	14m22s		**opt**	2h3m23s		**opt**
NHL13-1212-76	18m45s		**opt**	45m58s		**opt**	2h28m56s		**opt**	31h39m27s		**opt**
NHL11-1312-78	16m3s		**opt**	41m0s		**opt**	2h16m24s		**opt**	3h27m45s		**opt**
NHL11-1312-72	18m0s		**opt**	57m40s		**opt**	1h40m0s		**opt**	4h17m50s		**opt**
NHL13-1216-74	38m32s		**opt**	1h16m44s		**opt**	4h45m52s		**opt**	–		*feas*
NHL13-2416-78	–	–		4h23m27s		**opt**	–		*feas*	–		*feas*
NHL13-2416-74	2h40m32s		**opt**	5h39m17s		**opt**	16h29m31s		**opt**	–		*feas*
NHL11-2624-74	12h54m23s		**opt**	40h40m18s		**opt**	40h53m7s		**opt**	–	–	

Table 4.4 *Results for the partial-league instances with minimization of total travel distance* Running times and output solution status from CPLEX® on four different models. For a model in which a particular constraint has been relaxed, the reference to the constraint, C_i, is enclosed between brackets, $[\mathsf{C}_i]$, on the top of the table. The results are separated into two configurations of the revisit-gap parameters: long (*LGap*) and short (*SGap*). The solution status (*Sol*) indicates whether (*feas*) or not (–) a feasible solution, not necessarily the optimal (**opt**), has been found within the time limit of 48 hours.

	Model I			Model II			Model III			Model IV		
	C_1	C_2	C_3	C_1	C_2	C_3	C_1	C_2	C_3	$[\mathsf{C}_1]$	$[\mathsf{C}_2]$	$[\mathsf{C}_3]$
	C_4	$[\mathsf{C}_5]$	C_6	C_4	$[\mathsf{C}_5]$	C_6	$[\mathsf{C}_4]$	$[\mathsf{C}_5]$	$[\mathsf{C}_6]$	$[\mathsf{C}_4]$	$[\mathsf{C}_5]$	$[\mathsf{C}_6]$
	C_7	C_8	C_9	$[\mathsf{C}_7]$	$[\mathsf{C}_8]$	$[\mathsf{C}_9]$	$[\mathsf{C}_7]$	$[\mathsf{C}_8]$	$[\mathsf{C}_9]$	$[\mathsf{C}_7]$	$[\mathsf{C}_8]$	$[\mathsf{C}_9]$
Param. *LGap+MinDist*												
Instance	Time	*Sol*		Time	*Sol*		Time	*Sol*		Time	*Sol*	
NHL11-1208-76	–	–		–	–		–	*feas*		–	*feas*	
NHL13-1208-76	–	–		–	–		–	*feas*		–	*feas*	
NHL11-1208-74	–	–		–	–		–	*feas*		–	*feas*	
NHL13-1208-74	–	–		–	–		–	*feas*		–	*feas*	
NHL11-1208-72	–	–		–	–		–	*feas*		–	*feas*	
NHL11-1210-74	–	–		–	*feas*		–	*feas*		–	–	
NHL13-1212-76	–	–		–	–		–	–		–	–	
NHL11-1312-78	–	–		–	–		–	*feas*		–	–	
NHL11-1312-72	–	–		–	–		–	*feas*		–	–	
NHL13-1216-74	–	–		–	–		–	–		–	–	
NHL13-2416-78	–	–		–	–		–	–		–	–	
NHL13-2416-74	–	–		–	–		–	–		–	–	
NHL11-2624-74	–	–		–	–		–	–		–	–	
Param. *SGap+MinDist*												
Instance	Time	*Sol*		Time	*Sol*		Time	*Sol*		Time	*Sol*	
NHL11-1208-76	–	*feas*		–	*feas*		–	*feas*		–	*feas*	
NHL13-1208-76	–	*feas*		–	*feas*		–	*feas*		–	*feas*	
NHL11-1208-74	–	*feas*		–	*feas*		–	*feas*		–	*feas*	
NHL13-1208-74	–	–		–	*feas*		–	*feas*		–	*feas*	
NHL11-1208-72	–	–		–	*feas*		–	*feas*		–	*feas*	
NHL11-1210-74	–	*feas*		–	*feas*		–	*feas*		–	*feas*	
NHL13-1212-76	–	–		–	–		–	–		–	–	
NHL11-1312-78	–	*feas*		–	–		–	–		–	–	
NHL11-1312-72	–	*feas*		–	*feas*		–	*feas*		–	–	
NHL13-1216-74	–	–		–	–		–	–		–	–	
NHL13-2416-78	–	–		–	–		–	–		–	–	
NHL13-2416-74	–	–		–	–		–	–		–	–	
NHL11-2624-74	–	–		–	–		–	–		–	–	

Table 4.5 *Results for the 2012 instances with and without minimization of total travel distance* Running times and output solution status from CPLEX® on four different models. In each case, two configurations of the revisit-gap parameters, *LGap* and *SGap*, has been tested with (*MinDist*) and without minimization of distance. The solution status (*Sol*) indicates whether (*feas*) or not (–) a feasible solution, not necessarily the optimal (**opt**), has been found within the time limit of 48 hours.

	Model I			Model II			Model III			Model IV		
	C_1 C_4 C_7	C_2 $[C_5]$ C_8	C_3 C_6 C_9	C_1 C_4 $[C_7]$	C_2 $[C_5]$ $[C_8]$	C_3 C_6 $[C_9]$	C_1 $[C_4]$ $[C_7]$	C_2 $[C_5]$ $[C_8]$	C_3 $[C_6]$ $[C_9]$	$[C_1]$ $[C_4]$ $[C_7]$	$[C_2]$ $[C_5]$ $[C_8]$	$[C_3]$ $[C_6]$ $[C_9]$
Instance `NHL12-1315a`												
Parameters	Time		*Sol*	Time		*Sol*	Time		*Sol*	Time		*Sol*
LGap	4h17m36s		**opt**	2h13m11s		**opt**	4h24m18s		**opt**	2h39m58s		**opt**
SGap	15m56s		**opt**	9m4s		**opt**	55m20s		**opt**	41m10s		**opt**
LGap+MinDist	–		*feas*	–		*feas*	–		*feas*	–		*feas*
SGap+MinDist	–		*feas*	–		*feas*	–		*feas*	–		*feas*
Instance `NHL12-1315b`												
Parameters	Time		*Sol*	Time		*Sol*	Time		*Sol*	Time		*Sol*
LGap	–	–	–	–	–	–	–		*feas*	–		*feas*
SGap	–	–	–	8h59m43s		**opt**	–		*feas*	–		*feas*
LGap+MinDist	–	–	–	–	–	–	–		*feas*	–		*feas*
SGap+MinDist	–	–	–	–	–	–	–		*feas*	–		*feas*

Table 4.6 *Results for the 2012-13 season schedule of the Eastern Conference* Constraint violations and travel distance on the results from Table 4.5 on Model III for the instance `NHL12-1315a`. The column $\Delta Dist$ refers to the difference between the total travel distance for the corresponding solution and the travel distance for the actual NHL 2012-13 schedule of the Eastern Conference, which is 268034 miles.

Model III		Number of violations									
Parameters	*Sol*	C_1	C_2	C_3	$[C_4]$	$[C_5]$	$[C_6]$	$[C_7]$	$[C_8]$	$[C_9]$	$\Delta Dist$
LGap	**opt**	0	0	0	0	0	0	0	0	0	+40492
SGap	**opt**	0	0	0	0	0	0	0	0	0	+33661
LGap+MinDist	*feas*	0	0	0	18	0	9	0	0	2	+50244
SGap+MinDist	*feas*	0	0	0	2	0	0	0	0	0	+33344

Table 4.7 *Results for the entire-league instances without minimization of total travel distance* Running times and output solution status from CPLEX® on four models (different constraint relaxations) with one revisit-gap configuration (short). For a model in which a particular constraint has been relaxed, the reference to the constraint, C_i, is enclosed between brackets, $[\mathsf{C}_i]$, on the top of the table. The solution status (*Sol*) indicates whether (*feas*) or not (–) a feasible solution, not necessarily the optimal (**opt**), has been found within the time limit of 48 hours.

	Model I			Model II			Model III			Model IV		
	C_1 C_4 C_7	C_2 $[\mathsf{C}_5]$ C_8	C_3 C_6 C_9	C_1 C_4 $[\mathsf{C}_7]$	C_2 $[\mathsf{C}_5]$ $[\mathsf{C}_8]$	C_3 C_6 $[\mathsf{C}_9]$	C_1 $[\mathsf{C}_4]$ $[\mathsf{C}_7]$	C_2 $[\mathsf{C}_5]$ $[\mathsf{C}_8]$	C_3 $[\mathsf{C}_6]$ $[\mathsf{C}_9]$	$[\mathsf{C}_1]$ $[\mathsf{C}_4]$ $[\mathsf{C}_7]$	$[\mathsf{C}_2]$ $[\mathsf{C}_5]$ $[\mathsf{C}_8]$	$[\mathsf{C}_3]$ $[\mathsf{C}_6]$ $[\mathsf{C}_9]$
Instance	Time		*Sol*	Time		*Sol*	Time		*Sol*	Time		*Sol*
NHL09-00a	11h31m		**opt**	36h25m		**opt**	–		–	–		–
NHL10-00a	33h49m		**opt**	33h26m		**opt**	–		–	–		–
NHL11-00a	15h40m		**opt**	36h5m		**opt**	–		–	–		–
NHL13-00a	10h47m		**opt**	–		–	–		–	–		–
NHL09-04a	–		–	–		–	–		–	–		–
NHL10-04a	39h26m		**opt**	–		–	–		–	–		–
NHL11-04a	–		–	–		–	–		–	–		–
NHL13-04a	–		–	–		–	–		–	–		–
NHL09-08a	–		–	–		–	–		–	–		–
NHL10-08a	–		–	–		–	–		–	–		–
NHL11-08a	–		–	–		–	–		–	–		–
NHL13-08a	–		–	–		–	–		–	–		–

CHAPTER 5

AN ADAPTIVE LARGE NEIGHBORHOOD SEARCH

The results outlined in the preceding chapter support the general idea in the sports scheduling literature that a problem like the scheduling of the NHL games might require more specialized optimization approaches (Schaerf, 1999; Trick, 2011). Therefore, in this chapter, we propose an algorithm that integrates both classic and novel optimization techniques into an Adaptive Large Neighborhood Search (ALNS), which is well-suited for tightly-constrained problems and has provided outstanding results for transportation and other scheduling problems (Pisinger and Ropke, 2010; Kovacs *et al.*, 2012).

In the following sections, we describe our ALNS algorithm for the NHL scheduling problem, which includes a review of the heuristics introduced by Fleurent (1987) and computational experiments on setting the parameters of the algorithm, and present some results for several instances we have introduced in the preceding chapter.

5.1 Model

Here, we consider a model where the constraints enumerated in Chapter 3 for the NHL scheduling problem are all soft constraints. In addition to the essential constraints that, for example, does not allow more than one game for a team on one day, this model consists basically of an objective function defined by a weighted sum of the number of violations for the nine constraints, C_1, C_2, ..., C_9. Indeed, in this chapter, the *objective function value*, or simply *cost*, with regard to a solution (schedule) S for the NHL scheduling problem is given by

$$f(S) = \sum_{i=1}^{9} w_i f_i(S),$$

where $f_i(S)$ is the number of violations of the constraint C_i in the schedule S, and w_i is the corresponding non-negative weight (or penalty).

5.2 Algorithm

The Adaptive Large Neighborhood Search (ALNS) algorithm that we propose in this thesis for the NHL scheduling problem follows the framework introduced by Ropke and Pisinger (2006), which is an extension of the Large Neighborhood Search (LNS) given by Shaw

(1998). The ALNS is also essentially based on the *ruin and recreate* paradigm presented by Schrimpf *et al.* (2000).

Generally speaking, that framework is made up of repeated attempts to improve a certain solution, referred to as the *current solution*, which must initially be either provided or generated from scratch. Each of those attempts defines an iteration of the ALNS during which part of the current solution is modified and the resulting new solution is evaluated to be either rejected or accepted to become the "current" solution for the next iteration.

Indeed, during each ALNS iteration, two kinds of algorithms are systematically applied to the current solution: first, one that relax part of the solution, and then, another algorithm that re-optimizes the problem over the relaxed part in attempt to obtain a better whole solution. These partial-relaxation and re-optimization algorithms are well known in the ALNS literature as *destroy operator* and *repair operator*, respectively.

In general, an ALNS provides a number of operators, each one implementing a different strategy, either for destroying or for repairing a solution. As suggested by Ropke and Pisinger (2006), the approach might become more robust on the whole when alternating between different destroy and repair operators, as one operator may be more effective than another on different instances of the problem.

One destroy and one repair operators are selected at the beginning of each iteration in a random weighted manner according to their individual performance as the search progresses. The idea is to make the most promising operators more likely to be chosen (and applied) through the iterations of the ALNS.

In our ALNS, all the destroy and repair operators are heuristics. Every destroy operator set games *free* by canceling some game-day assignments in the current solution, or equivalently, "removes" certain games from the current schedule. In the same way, every repair operator assigns a day to each free game, or equivalently, "inserts" the free games back into the current schedule—hopefully on different (and better) days than before. This might explain why, from an application point of view, the literature often refers to the destroy and repair operators as *removal* and *insertion* operators, respectively.

We now present the design decisions on the solution acceptance criteria, and on the selection of operators and the adjustment of their weights, which control the master level of the ALNS. In the next subsections, we then address a multi-heuristic approach to construct an initial solution for the problem, and describe the operators that we propose either for partially destroying an NHL schedule or for repairing it to a complete schedule.

5.2.1 Controlling the master level of the algorithm

Basically, our ALNS algorithm consists of the construction of an initial solution (S_{init}) and a loop that, while not all constraints of the problem are satisfied, tries to improve the current solution (S_{curr}) through a fixed number of iterations.

Here, we first present the solution acceptance criteria for each iteration of the algorithm; and then, we describe the strategies for the adaptive layer of the search, which controls the choice of the operators according to their individual performance in past iterations. An outline, with a pseudo-code, of the master level of the algorithm is presented in Appendix A.1.

Solution acceptance criteria

We have chosen to use the acceptance criteria from Simulated Annealing, which has been widely used in the ALNS literature. Therefore, during each iteration of our ALNS, after a destroy and a repair operators have been applied, the new solution is accepted whenever its cost is not worse (not higher) than the cost of the current solution; and otherwise, the new solution is accepted with a certain probability. The worse the new solution is, the less likely it is to be accepted. In addition, this probability is proportional to a control parameter known as the *temperature*, which is slightly decreased as the ALNS advances.

To be more precise, we initially set an initial temperature T_0 to a high value, and then, for each iteration k of the ALNS, the temperature is lowered according to a *cooling schedule* defined by $T_{k+1} = c\,T_k$, where c, such that $0 < c < 1$, is known as the *cooling factor*. If S_{curr}, with cost $f(S_{curr})$, is the current solution, and S_{temp}, with cost $f(S_{temp})$, is the temporary solution generated by the corresponding pair of destroy and repair operators, then S_{temp} is accepted whenever the condition $f(S_{temp}) \leq f(S_{curr})$ holds; otherwise, the temporary solution is accepted with probability $e^{-(f(S_{temp})-f(S_{curr}))/T_k}$, which depends on the current temperature, T_k, and on the corresponding increase in the cost, $f(S_{temp}) - f(S_{curr})$.

Using the strategy of Ropke and Pisinger (2006), instead of providing the initial temperature as a parameter to the ALNS, a more meaningful parameter, the *start temperature control parameter* (which we denote by τ), is used to calculate the temperature according to an evaluation of the initial solution at hand.

Indeed, we first evaluate the initial solution, S_{init}, with a slightly modified cost function, $\hat{f}$, that is also a weighted sum of the constraint violations, but without the terms corresponding to the first two constraints (the "most costly" ones). And then, the initial temperature is set to a value such that, at the beginning of the algorithm, a temporary solution causing a cost increase of exactly τ% of $\hat{f}(S_{init})$ would be accepted with probability 0.5.

In addition, because we have chosen to run the ALNS for a fixed and relatively large

number of iterations, we also compute the cooling factor, instead of following the strategy by Ropke and Pisinger (2006), which provides the cooling factor as another parameter to the algorithm. In fact, we set the cooling factor, c, which controls the constant rate of decrease in the temperature parameter to a value such that a temporary solution one unit worse than the current solution would have only 1% of chance of being accepted at the last ALNS iteration. The cooling factor depends thus on the initial temperature and on the (typically large) maximum number of ALNS iterations, and its value is usually very close to 1.0. This results in only a slight decrease in the temperature from one iteration to the next.

Operator selection

As previously mentioned, during each iteration of the ALNS, two operators are selected: one for destroying part of the current solution and another for repairing it. The selection of operators is implemented by first assigning to each operator a weight that represents its past performance on the problem instance being solved. Then that weight is used to associate a probability of selection with each individual operator. In fact, the probability of an operator being chosen is proportional to its weight, as the ALNS uses a *roulette wheel selection*.

To be more precise, if ω_i is the weight of a destroy operator i, its probability of being selected is $\omega_i / \sum_{o \in I^-} \omega_o$, where I^- is the set of destroy operators provided to the ALNS. The strategy for selecting a repair operator is identical to that of a destroy operator.

Weight adjustment

During the search, if all the operators (either for destroying or for repairing) had the same weight, then every operator would have, at any time, the same probability of being selected. The ALNS, however, has an adaptive layer that automatically adjusts the weights associated to operators in order for those weights to mimic the performance of the operators in earlier iterations.

In our implementation, as suggested by Ropke and Pisinger (2006), the weights are indeed systematically recalculated at the end of every certain number of iterations. The whole search is split into blocks of consecutive iterations called *segments*. All the segments have the same number of iterations, which defines the *segment size*, a parameter that is provided to the ALNS.

According to the performance of the operators during a segment, a *score* is assigned to each of them. At the end of a segment, the weight of every operator (and thus its selection probability) is then computed as a weighted sum of its score during the segment and its overall score since the beginning of the search.

To be more precise, let s be an ALNS segment, and let o be an operator that is selected at least once through the iterations of the segment s. In order to compute the weight of the operator o for the next segment, a score π_o is set to zero at the beginning of the segment s and might then be increased by a certain amount each time the operator o is applied through s. The increment amount depends on the quality of the (temporary) solution obtained after the corresponding destroy and repair operators have been applied.

We have chosen to use three *score adjustment parameters*, σ_{bes}, σ_{bet}, and σ_{acc}, to specify the amounts by which the score may be increased. Indeed, at any iteration when o is applied during the segment s, the score π_o increases by σ_{bes} if the cost of the resulting solution is the best (lowest) found since the beginning of the search; otherwise, π_o increases either by σ_{bet} if the resulting solution cost is better than the cost of the current solution, or by σ_{acc} if the resulting solution is accepted by the ALNS but its cost is worse than the cost of the current solution. The score is not changed if the resulting solution is not accepted.

When the end of the segment s is reached, and the operator o had been applied θ_o times and gained a total of π_o as score during that segment, the weight of o for the next segment is computed as $\omega_o\,(1-\rho) + \rho(\pi_o/\theta_o)$, where ρ, such that $0 \leq \rho \leq 1$, is the *reaction factor* controling the degree of change on the current weight, ω_o, in response to the "smoothened" score obtained by the operator o.

5.2.2 Constructing an initial solution

In order to provide a complete NHL schedule at the beginning of the ALNS algorithm, we propose a construction approach that implements the heuristics introduced by Fleurent (1987). They were originally intended to be used in an interactive manner by an expert scheduler that would call any of the corresponding procedures whenever needed during the scheduling process. But Fleurent (1987) also proposes an implementation that calls the heuristic procedures in a specific order, without any interaction with the user. Since the game-day assignments in Fleurent's heuristics need to satisfy most of the constraints of the problem, that approach alone may not be able to fully schedule the free games (at least not for the hardest instances, with none or too few extra arena dates per team). We are therefore proposing to construct an initial NHL schedule by first applying those heuristics as suggested by Fleurent (1987), and then, if there still are free (missing) games, by scheduling them with a basic greedy approach that evaluates the incremental cost associated with every possible day for each game being scheduled. This might then yield a reasonable NHL schedule from which to start our search for an improved solution.

Fleurent's heuristics will be individually described in the next subsections (and outlined in Appendix B.1). Before that, we outline the whole construction approach by the steps

informally stated below, where the heuristics we mention refer to those introduced by Fleurent (1987) and, to stay in line with their original conception, a "feasibly-possible scheduling" is a game-day assignment that satisfies all the constraints in the set C_F, except perhaps C_5 on the minimum number of games per week, where $\mathsf{C}_F = \{\mathsf{C}_0, \mathsf{C}_1, \mathsf{C}_2, \mathsf{C}_4, \mathsf{C}_5, \mathsf{C}_6\}$.

Step 1. *Schedule forced trips.* Apply the *forced-trip heuristic* to schedule as many free games as feasibly possible by first identifying long periods of days with no arena dates for a team, and then, by assigning the respective team to visit, in a single trip, at most seven of its distant opponents exclusively during the corresponding period of arena unavailability.

Step 2. *Schedule forced home games.* Apply the *forced-home heuristic* to schedule, for each trip lasting more than one week, two home games for the respective traveling team: one game on its last (latest) arena date before the trip, and the other, on its first arena date after the trip.

Step 3. *Schedule free trips.* Apply the *free-trip heuristic* to schedule all current free games opposing teams based far from each other by first identifying, for each team, at most three of its distant opponents to be visited in a single trip, and then, by scheduling every one of those long-distance trips on a period that does not violate the constraints in C_F and that contains the minimum possible number of arena dates of the visiting team.

Step 4. *Schedule forced home games.* Apply the *forced-home heuristic* to have, for each trip lasting more than seven days, a home game scheduled on the last and on the first arena day of the traveling team before and after that trip, respectively.

Step 5. *Schedule weekend games.* Apply the *weekend-game heuristic* to schedule as many free games as feasibly possible respectively on Saturdays, on Fridays, and on Sundays, by trying not to schedule a team to play away on a weekend containing some of its own arena dates, and by never putting a team to play an away game within a period when it has already been scheduled to visit distant opponents.

Step 6. *Schedule weekday games.* Apply the *weekday-game heuristic* to schedule as many free games as feasibly possible on weekdays from Monday through Thursday, by trying not to schedule a team to play away on its own arena dates, and by never assigning days from periods of long-distance visits for the teams involved in the game being scheduled.

Step 7. *Exchange games.* Apply the *single exchange heuristic*, and then the *double exchange heuristic*, to schedule as many free games as feasibly possible by allowing some of the previously scheduled short-distance visits to be rescheduled on alternative days.

Step 8. *Schedule the remaining free games.* Select at random every remaining free game and schedule it in a greedy manner on a day that, besides not violating the constraint C_0 (no more than one game a day for a team), produces the lowest incremental cost.

We now describe each of the heuristics initially proposed by Fleurent (1987). To anticipate, they have been adapted as some of the operators for our ALNS, which is described later in this chapter. For a more precise description of each of them, which outlines the steps they perform, see Appendix B.1.

Forced-trip heuristic

The forced-trip heuristic tries to generate as many trips as feasibly possible, each one made up exclusively of games taking place during a long period of consecutive days with no arena dates for the respective traveling team. The overall idea is straightforward. In order to preserve dates for eventual subsequent allocations of home games by other heuristics, no team is scheduled to play away on days when its own arena is available. Furthermore, in the course of any trip being generated, the visiting team only plays against several distant opponents, which in turn, should ideally be located in the same geographical area. This aims to reduce the total distance traveled by the teams during the entire regular season.

The strategy on preserving home dates is directly related to *forced trips*, which are trips taking place entirely during a period when the arena of the traveling team is not available (Ferland and Fleurent, 1991). Because the schedule is intended to be feasible with regard to the constraints in C_F (including the arena availability constraint), when applying the forced-trip heuristic, the only games scheduled for a team between two of its consecutive arena dates are then games to be played away. Moreover, the corresponding opponents on those games must be located farther than a given threshold distance (900 miles) from the arena of the traveling team and, as we have anticipated, they should be based in the same geographical area.

In an attempt to satisfy this goal, the following saving measure is used when selecting the teams to be visited during a forced trip:

$$s_a(h_1, h_2) = dist(h_1, a) + dist(a, h_2) - dist(h_1, h_2),$$

where a is the traveling team, h_1 and h_2 are two potential consecutive away opponents, and $dist(\cdot, \cdot)$ is the distance between the corresponding pair of teams.

A large value for this measure, which was used by Clarke and Wright (1964) in a classical algorithm for vehicle routing problems, suggests that indeed the two corresponding opponents are located close to each other when compared to their distances to the arena of the traveling

team.

Basically, the heuristic starts building a new trip whenever it finds a team that can visit two distant opponents on a single two-game trip scheduled between two of its consecutive arena dates (which would rather be widely separated from each other) without violating the constraints in C_F. Then, it attempts to extend the trip for no more than seven games scheduled only to the current period of arena unavailability for the traveling team. Indeed, the heuristic tries to schedule that same team to directly visit other distant opponents both before and after the (so far two-game) trip. Each away game appended to the trip is selected as to maximize the saving measure defined above, and the constraints in C_F must be satisfied all the time.

Forced-home heuristic

The forced-home heuristic tries to schedule home games to the begin and to the end of every long-lasting trip for a team. Indeed, the goal is to schedule, for each trip lasting more than seven days, the respective traveling team to play home at its first arena date just before the trip and at its first arena date after the trip. Every game-day assignment must satisfy the constraints in C_F. The heuristic seeks, as a priority, to schedule games for which the arena of the visiting team is not available at the day being assigned; and in case of tie, the game corresponding to the visit of the nearest opponent is chosen to be scheduled.

Free-trip heuristic

The free-trip heuristic, as the forced-trip heuristic, generates for each team, trips that are made up exclusively of games to be played against distant opponents. In this case, however, trips are individually generated by first determining a short promising sequence of away games, and only then, by trying to assign suitable days to the respective games. In other words, once an appropriate new trip is identified, the heuristic is "free" to choose any promising period when it must take place, with the condition that the constraints in C_F are not violated.

The overall goals when generating every free trip are twofold: *(1)* to schedule a team to visit distant opponent located nearby in a same area so as to reduce the total travel distance; and *(2)* to preserve as many arena dates of the traveling team as possible, even though such dates are now allowed to be taken, if necessary.

At first, a team a having the highest number of distant (more than 900 miles) away games remaining to be scheduled is chosen as the traveling team. Because it may be too hard to find a suitable period of days for a trip with many games, the free trips are made up of only

two or only three games. The first two distant opponents, h_1 and h_2, to be visited by a in the free trip, $r(a)$, being generated are chosen as to maximize $s_a(h_1, h_2)$, the saving defined earlier. A third opponent, h_3, is then appended to $r(a)$ if the value $s_a(h_3, h')$ is maximized with $h' = h_1$ or with $h' = h_2$, among all distant opponents that a must still be scheduled to visit. Finally, the heuristic tries to schedule the free trip $r(a)$ to a period having the least number of arena dates for the traveling team, a.

Weekend-game heuristic

The weekend-game heuristic tries to schedule as many of the short-distance visits as possible exclusively to weekend days as this is assumed to be a promising factor to increase spectator attendance at home games. Every game-day assignment satisfies the constraints in C_F and no game is scheduled to a period when either of the respective teams has already been assigned to visit distant opponents.

The heuristic is based on a specific set of "target days", which is successively characterized (in this order) by all Saturdays, all Sundays, and all Fridays of the regular season period. For each of those sets, the heuristic seeks, whenever possible, to select a target day and a free game that yield a feasible game-day assignment where two non-distant teams oppose each other without involving any long-distance trip period for either of them.

The actual assignments are, furthermore, based only on the arena availability of the visiting team during the respective weekend. Indeed, at any time, a game is scheduled exclusively on a target day when the arena of the visiting team, with regard to the corresponding three weekend days, is either *(i)* completely unavailable, *(ii)* available only on one day which is not the target day, *(iii)* available only on the target day itself, or *(iv)* available only on the other two (non-target) days. These four required conditions define the "levels of preference" that are successively taken into account when selecting game-day pairs for the assignments; on each of those levels, the goal is to carry out as many feasible assignments satisfying the corresponding condition as possible.

Weekday-game heuristic

The weekday-game heuristic tries to schedule the remaining free games to weekdays from Monday through Thursday. As for the preceding heuristic, the game-day assignments satisfy the constraints in C_F and no game is scheduled to a period when either of the respective teams has already been assigned to visit distant opponents.

In order to avoid wasting arena dates by scheduling a team to play away when its own arena is available, the heuristic adopts a strategy that is based on the Vogel Approximation

Method (Reinfeld and Vogel, 1958), which is a well-known simple procedure for obtaining a feasible solution to the classical transportation problem.

Indeed, a game-day assignment cost is initially calculated for each pair of free game and corresponding feasible day (when, in particular, the arena of the home team is available). This cost is directly proportional to the number of teams for which the respective arena is available at the same day. Furthermore, a large penalty is added to the cost whenever that day is also available for the visiting team. Subsequently, the two less costly potential assignments for each game are taken and the difference between their costs is computed. This difference between the two lowest cost for every free game can be seen as a measure of the current "regret" for failing to schedule the game on its less costly feasible day. Finally, the heuristic selects a game with the largest regret and assigns to it the corresponding feasible day having the smallest cost. After each assignment, the costs and the regrets are updated for all the free games involving either of the just-scheduled teams and for which that same day is also implicated in one of their respective (two) less costly potential assignments.

Exchange heuristics

Two exchange heuristics are presented in Fleurent (1987). Both heuristics try to schedule the free games by allowing some of the previously scheduled games to be rescheduled on alternative days.

Whenever possible, free games are scheduled on days that violate no constraints in C_F. However, for each iteration when no feasible day is available, the approach considers only the days that have already been assigned to other games involving either of the two teams in the current free game. One of those days is then selected if by just removing some of its previously scheduled game, it becomes feasible for the current game scheduling (in particular, the home and the away teams would still have only a single game to play on the chosen day). Every time this happens, the heuristic replaces the corresponding scheduled game on the selected day by the free game at hand and immediately tries to reschedule the removed game on another feasible day. This defines a new *recursive level* of the approach. If no feasible day is found for the removed game, the heuristic backtracks to the preceding level, reschedules that game on its last cancelled day, and tries to schedule the original free game on another day. For practical reasons, a parameter might be provided to specify the maximum number of levels that the heuristic can go through when attempting to schedule a free game.

In fact, it may also happen that the most promising day for a certain game scheduling by exchanges is a day when both the home and the away teams in the current free game have already been scheduled to play against other opponents. From this fact arises the difference between the two proposed exchange heuristics: while one allows two games to be

simultaneously cancelled from a same day, the other does not. Indeed, if no feasible day is already available, the called *single exchange heuristic* tries to find a day from which removing only one game would turn that day into a feasible candidate for the current free game to take place. On the other hand, whenever that same strategy fails, the called *double exchange heuristic* seeks to carry out the scheduling of the game by removing from a certain day exactly two games involving the teams in the current free game if it then yield a feasible assignment. Both heuristics backtrack when either no promising day is found or a given maximum recursive level is reached.

We now propose a number of destroy operators to be provided to the ALNS for the NHL scheduling problem.

5.2.3 Partially destroying a solution

In this subsection, we propose 15 operators to partially destroy the schedules through the iterations of the ALNS for the NHL scheduling problem. Each of these operators is a heuristic that cancels a given number of game-day assignments in the current schedule. What makes the heuristics different from each other are the criteria for choosing the games to be removed from the schedule. In particular, when selecting its target games, a removal operator may use a specific level of greediness (or randomness); and some operators may, for example, adopt a more systematic approach of destruction than others.

Here, we divide the destroy operators into five groups, as follows:

1. random-based removal
   - 01$^-$ *Random-based removal*
2. structure-based removals
   - 02$^-$ *Divisional removal*
   - 03$^-$ *Intradivisional removal*
   - 04$^-$ *Interdivisional removal*
   - 05$^-$ *Conferential removal*
   - 06$^-$ *Intraconferential removal*
   - 07$^-$ *Interconferential removal*
3. neighboring-based removals
   - 08$^-$ *Day-neighboring removal*
   - 09$^-$ *Place-neighboring removal*
4. critical removals

- $\mathsf{10}^-$ *Arena-critical removal*
- $\mathsf{11}^-$ *Sequential greedy removal*
- $\mathsf{12}^-$ *Cost-critical removal*

5. trip-based removals
   - $\mathsf{13}^-$ *Short-trip removal*
   - $\mathsf{14}^-$ *Close-trips greedy removal*
   - $\mathsf{15}^-$ *Scattered-trip removal*

In general, the amount that an operator is allowed to destroy in a solution has substantial impact on the quality of the solutions produced by an ALNS algorithm (Pisinger and Ropke, 2007). In particular, for most of the operators we propose here, the number of game-day assignments satisfying the selection criteria of the operator being applied to the schedule at hand might be too high for all them to be removed and eventually reinserted (by some repair operator) in an effective manner. We introduce then the following strategy, which comparing to the seminal work by Ropke and Pisinger (2006), can be seen as an intensification mechanism that is being integrated into the ALNS framework. Except for the operators $\mathsf{10}^-$, $\mathsf{11}^-$, $\mathsf{12}^-$, and $\mathsf{13}^-$, all the destroy operators will first choose a certain number of game-day assignments in the current schedule and try to evaluate the advantage of cancelling each of them, prior to proceeding with a rather few game removals.

The choice of the target assignments, which are the candidates to be eventually cancelled by the corresponding operator, is based on the specific strategy of each heuristic. On the other hand, when actually removing games from a schedule, the same greedy-based strategy is used by all those operators. This strategy consists in given a higher priority to target-assignment cancellations that individually leads to the best (maximum) reduction on the objective function value; and, in case of tie, the arena dates of the respective home and away teams are also taken into account.

To be more precise, the cancellation of a game-day assignment will rely on a function that combines, in a weighted manner, both the incremental cost of the cancellation and the effectiveness of that assignment for the corresponding home and away teams regarding their arena availabilities. Let that function be denoted by f_- and let its value for removing an assignment $[a@h\colon d]$ from a schedule S be defined as

$$f_-(a,h,d) = 5\, f_-^{\Delta}(a,h,d) + \bar{A}_{a,h,d},$$

where $f_-^{\Delta}(a,h,d)$ is the corresponding *removal incremental cost*, namely, $f(S \setminus \{[a@h\colon d]\}) - f(S)$; and $\bar{A}_{a,h,d}$, which we refer to as the *arena-utility value* for having the day d assigned

to the game $a@h$, is a constant given by

$$\bar{A}_{a,h,d} = \begin{cases} 1 & \text{if } d \text{ is provided by } a \text{ but not by } h; \\ 2 & \text{if } d \text{ is provided neither by } a \text{ nor by } h; \\ 3 & \text{if } d \text{ is provided both by } a \text{ and by } h; \\ 4 & \text{if } d \text{ is provided by } h \text{ but not by } a. \end{cases}$$

The strategy is then to remove from a schedule a given number of target games having the lowest values of the function f_{-}. Obviously, this is an attempt to cancel assignments causing to most violations of constraints, while trying to release arena dates for teams that have been scheduled to play away in the current schedule when their own arena is available.

In the remainder of this subsection, we describe every one of the destroy operators enumerated above. Suppose that any of those operators is a procedure to which at least the following three parameters are given: S, a complete schedule (from the current ALNS iteration); $\hat{n}$, the maximum number of game-day assignments to be selected from S as candidates for the cancellations; and $\tilde{n}$, the number of assignments to be actually cancelled in the schedule S. Consider also that any complete schedule is made up of exactly n game-day assignments, and the condition $0 < \tilde{n} < \hat{n} \leq n$ holds (otherwise, if $\tilde{n} \geq \hat{n}$ then all the corresponding target assignments could be cancelled with no need for evaluating them beforehand).

In addition to the descriptions that follows, Appendix C.1 outlines the steps performed by each destroy operator.

Random-based removal

At first, we propose a destroy operator in which the target games are chosen at random among all games in the current schedule. Indeed, the *Random-based removal* ($\mathsf{01}^{-}$) is a heuristic that starts by choosing at random $\hat{n}$ target assignments in the current schedule, evaluates by f_{-} the extended incremental cost of each of them, and then cancels $\tilde{n}$ target assignments having the lowest values of f_{-}.

Structure-based removals

We now propose six destroy operators in which the target games for each operator are chosen at random among a specific subset of games that is determined only by the structure of the NHL. In particular, such subset does not depend on the game-day assignments in the current schedule. As for the preceding operator, once $\hat{n}$ target games have been chosen, those games are evaluated by f_{-}, and the $\tilde{n}$ target games in S yielding the lowest extended incremental cost are removed.

The first three of these operators choose their $\hat{n}$ target games by taking into account the NHL divisions of the home and away teams in each game. Namely, the *Divisional removal* ($\mathsf{02}^-$) removes only games individually involving at least one team from a division that is randomly chosen beforehand; the *Intradivisional removal* ($\mathsf{03}^-$) removes only games individually involving both home and away teams from a division that is also randomly chosen in advance; and the *Interdivisional removal* ($\mathsf{04}^-$) removes only games individually involving teams from two divisions that are also chosen at random when the operator begins.

The other three structure-based destroy operators choose their $\hat{n}$ target games by taking into account the NHL conferences of the home and away teams on each game. Indeed, the *Conferential removal* ($\mathsf{05}^-$) removes only games individually involving at least one team from a conference that is chosen beforehand; the *Intraconferential removal* ($\mathsf{06}^-$) removes only games individually involving both home and away teams from a conference that is also randomly chosen in advance; and the *Interconferential removal* ($\mathsf{07}^-$) removes only games individually involving teams from different conferences.

Neighboring-based removals

We now propose two destroy operators in which the target games are chosen either by a neighboring area where they are scheduled to take place or by a neighboring day when they are planned to occur.

At first, both operators randomly choose a time-window defined by a certain number of consecutive days in the schedule. The $\hat{n}$ target game-day assignments are then selected from within that time-window only. However, while the target of the *Day-neighboring removal* ($\mathsf{08}^-$) is made up by all the games assigned to that period of days, the target assignments in the *Place-neighboring removal* ($\mathsf{09}^-$) are, in addition, restricted to games to be played within a geographic region that is also chosen at random each time the operator is applied. Finally, both operators cancel the $\tilde{n}$ target games in S with the lowest values of f_-.

Critical removals

We now propose three destroy operators where the target games are chosen by some more careful evaluation which can involve incremental costs, arena-utility values, and conflicting games for promising reassignments. Here, given a constraint C_i, with $i \in \{0, 2, 3, 4, 6, 7, 8, 9\}$, we say that two games in a certain schedule are C_i*-conflicting* games if together they are involved in some violation for C_i.

One of these operators removes a given number of games from the schedule in a sequentially greedy fashion. Indeed, at each iteration, the *Sequential greedy removal* ($\mathsf{11}^-$) selects

and removes from the schedule the current worst game-day assignment regarding the incremental cost of its cancellation (the lower, the worst) and, in case of tie, also regarding its arena-utility value (the lower, the worst).

The other two critical destroy operators, *Arena-critical removal* ($\mathsf{10}^-$) and *Cost-critical removal* ($\mathsf{12}^-$), can also be seen as iterative procedures. In this case, however, each iteration is made up of three basic steps. In the first step, a poorly assigned game (regarding the specific evaluation adopted by each operator) is selected and removed from the schedule. Then, in the second and third steps, a promising day for the just-removed game is chosen, and some other assignments may also be cancelled as an attempt to turn that day into a better candidate for a possible eventual reassignment to the game removed in the first step.

Indeed, the operator $\mathsf{10}^-$ cancels, during the first and second steps, two assignments having the worst arena-utility value, such that the visiting team of the game removed in the second step is the host team of the game removed in the first one. In addition, the second game is removed from a promising day for the first-removed game, a day that, in particular, satisfies the constraint on the minimum gap for revisits (C_6). On the other hand, the operator $\mathsf{12}^-$ cancels, in the first step, the worst current assignment regarding the incremental cost and (in case of tie) also regarding the arena-utility value. Then, in the second step, this operator identifies a day that does not violate the constraint on the minimum gap for revisits (C_6) and that has the lowest number of conflicting games for the game removed in the first step. Finally, in the third step, both operators use an anticipation strategy that removes up to two C_i-conflicting games (if any) for the current potential reassignment (which is entirely hypothetical, as these are operators for removals only), regarding every constraint C_i, with $i = 0, 1, 2, 3, 4$, in this order.

Trip-based removals

Finally, we propose three destroy operators where the choice of the target games is based on the trips induced by the current schedule.

The first of these operators, the *short-trip removal* ($\mathsf{13}^-$), can be seen as an iterative procedure where each iteration is made up of three basic steps. In the first step, a trip having the least number of games is chosen and all its games are removed from the schedule. Regarding each of those games, the operator identifies, in the second step, a promising day (out from the former trip period) for a potential reassignment that satisfies the constraint on the minimum span for revisits (C_6) and that has the lowest corresponding number of conflicting games. In case of tie, the choice of that day is also given by the minimum incremental cost on the objective function. Then, in the third step, the same anticipation strategy used by the preceding (critical) operator is applied in order to remove up to two

conflicting games (if any) for each potential reassignment.

The other two trip-based removals explicitly create a set of $\hat{n}$ target game-day assignments before removing $\tilde{n}$ of them in a greedy fashion with regard to their respective incremental cost and (in case of tie) arena-utility values. At the beginning, however, both operators identify a period of days from which the target games are chosen. Such period of days is somehow related to poorly-defined trips. Indeed, the *close-trips removal* ($\mathsf{14}^-$) initially selects the two consecutive trips (for the same team) having the lowest number of days between them. On the other hand, the *scattered-trip removal* ($\mathsf{15}^-$) selects, at the beginning, a long-lasting trip (defined by a given threshold) that has relatively few games. In the case of the operator $\mathsf{14}^-$, the target games are selected by the minimum arena-utility value among all the games scheduled during the period of the two corresponding chosen trips and also during the days between them. In the operator $\mathsf{15}^-$, the target games are also chosen by the minimum arena-utility value, but they are selected among all the games scheduled during the period of the "scattered" trip chosen at the start. Finally, both operators cancel the $\tilde{n}$ target games having the lowest values of f_-.

We now propose some repair operators to be provided to the ALNS for the NHL scheduling problem.

5.2.4 Repairing a partial solution

In this subsection, we propose five operators to repair the partial schedules through the iterations of the ALNS for the NHL scheduling problem. Every operator is a heuristic that, at each ALNS iteration, is able to insert the free games into the current partial schedule.

Here, we say that a day d is a *free day* for a game $a@h$ if neither team a nor team h is currently scheduled to play on day d of the schedule at hand.

We divide the repair operators into four groups, as follows:

1. greedy-based insertions

   $\mathsf{01}^+$ *Single-evaluation greedy insertion*

   $\mathsf{02}^+$ *Updated-evaluation greedy insertion*

2. regret-based insertions

   $\mathsf{03}^+$ *Max-regret insertion*

3. exchange-based insertion

   $\mathsf{04}^+$ *Single-exchange insertion*

4. Fleurent's approach-based insertion

   $\mathsf{05}^+$ *Fleurent's approach-based insertion*

Except for the operators $\mathsf{03}^{+}$ and $\mathsf{05}^{+}$, the assignment of days to free games will rely on a function that combines, in a weighted manner, both the insertion incremental cost and the effectiveness of the assignment for the respective home and away teams with regard to their arena availabilities. Let that function be denoted by f_{+} and let its value for the potential scheduling of a game $a@h$ on a day d of a schedule S be defined as

$$f_{+}(a, h, d) = 5\, f_{+}^{\Delta}(a, h, d) - \bar{A}_{a,h,d},$$

where $f_{+}^{\Delta}(a, h, d)$ is the corresponding *insertion incremental cost*, namely, $f(S \cup \{[a@h\colon d]\}) - f(S)$; and $\bar{A}_{a,h,d}$ is the *arena-utility value* for having the day d assigned to the game $a@h$, as defined in Subsection 5.2.3.The strategy is then to schedule free games on free days leading to the lowest values of the function f_{+}. Obviously, this is an attempt to favor the assignments leading to the least violations of constraints, while trying to save arena dates for teams that are being scheduled to play away when their own arena is available.

In addition to the descriptions that follows, Appendix D.1 outlines the steps performed by each repair operator.

Greedy-based insertion

The first repair operator that we propose is a heuristic where free games are successively chosen at random and scheduled in a greedy manner to free days. The operator, which we refer to as *Single-evaluation greedy insertion* ($\mathsf{01}^{+}$), can be seen as a two-phase approach that first tries to schedule as many games as possible exclusively to days when the arena of the respective home team is available, and then schedules the possible remaining free games to other days on the current schedule. In both phases, days are assigned to games in the following way. At the beginning, the extended incremental cost determined by f_{+} is evaluated for each pair of free game and corresponding free day on the current partial schedule. Then, free games are sequentially scheduled to their respective best free days regarding those costs. So, every time that a game is chosen, the heuristic immediately seeks to find (and to assign) a corresponding free day with the lowest cost that was determined at the beginning.

The second repair operator is also a greedy-based heuristic. In this case, however, free games are successively chosen in decreasing order of the number of free games to be played at home for each team. Moreover, every time a free game is chosen, the heuristic seeks to assign to it the corresponding best free day on the current (updated) schedule. This operator, which we refer to as *Updated-evaluation greedy insertion* ($\mathsf{02}^{+}$), can also be seen as a two-phase approach that first tries to schedule as many games as possible exclusively to days when the arena of the respective home team is available, and then schedules the possible

remaining free games to other days in the current schedule. In both phases, free games are successively scheduled to their respective best free days, which leads to the lowest extended incremental cost at the moment of the assignment.

Regret-based insertion

Our third repair operator, which we refer to as *Max-regret insertion* ($\mathsf{03}^{+}$), is a regret-based heuristic where the first goal consists in avoiding assignments that "waste" arena dates for the respective away teams. As for the previous repair operators, the insertions rely on a specific evaluation to determine the cost of each possible game-day assignment. However, the main factor in this evaluation, which also involves the corresponding incremental cost, refers to a large penalty if a team was scheduled to visit an opponent on a day when its own arena is available. Generally speaking, the idea is to schedule the free games in decreasing order of the "regret" for failing to assign the first less costly day to a game and eventually having to assign its second less costly day instead. Although this is essentially identical to the weekday games heuristic described on Subsection 5.2.2, the possible assignments are evaluated in a rather refined manner by taking the change in the objective function value into account and by also considering the number of games that are free for the corresponding day at the time of the evaluation.

To be more precise, the cost, in operator $\mathsf{03}^{+}$, of scheduling a free game $a@h$ on a corresponding arena-feasible date d that is free in a partial schedule S is defined as follows:

$$f_{+}^{A}(a,h,d) = M_1\, A_{a,d} + M_2\, f_{+}^{\Delta}(a,h,d) + |\tilde{G}(d)|,$$

where M_1 and M_2 are large constants, such that $M_1 \gg M_2$; $A_{a,d}$ is equal to 1 if the day d is also provided by the away team a, and equal to 0 otherwise; $f_{+}^{\Delta}(a,h,d)$ is the incremental cost of assigning d to $a@h$, namely, $f(S \cup \{[a@h\colon d]\}) - f(S)$; and $|\tilde{G}(d)|$, which can be seen as a "competitiveness" factor for d, is the current number of free games that are arena-feasible (it would satisfy C_1) for the day d.

Exchange-based insertion

We now propose a repair operator that sequentially chooses the free games in decreasing order of the number of corresponding visits to be scheduled and try to schedule them by single exchanges. Although essentially similar to the single-exchange heuristic addressed in Subsection 5.2.2, we consider three fundamental modifications when selecting a promising day for the free game at hand. The first modification consists in preventing exchanges with games that were already scheduled before the heuristic started. This aims to stay in line with

the idea that, during each iteration of the ALNS framework, the corresponding non-destroyed part of the solution remains fixed. The second modification refers to the range of the days that are evaluated for the free games when no feasible day is available. Now, a candidate is a day that would allow a feasible assignment by first removing a single game currently scheduled on any day (possibly other than the candidate itself). Thus, the difference here is that a conflicting assignment with regard to any constraint is now taken into account for a possible exchange (rescheduling), instead of being restrict to an assignment that would only violate C_0. The third modification refers to the selection of a day. Whenever a tie occurs (where for each candidate day there is only one conflicting game in the schedule), the arena-utility values of both the intended assignment and the corresponding conflicting game in the schedule are taken into account.

Indeed, the *Single-exchange insertion* ($\mathsf{04}^+$) selects all free games, one at a time, in decreasing order of their quantity for the same away-home pair of teams. For each selected free game $\tilde{a}@\tilde{h}$, the heuristic proceed as follows. If feasible days are available, $\tilde{a}@\tilde{h}$ is scheduled on a feasible day $\hat{d}$ leading to the highest arena-utility value, $\bar{A}_{\tilde{a},\tilde{h},\hat{d}}$. Otherwise, the heuristic only considers as candidates those days on which scheduling $\tilde{a}@\tilde{h}$ would result in a single conflicting game somewhere in the current schedule. Among such candidates, $\tilde{a}@\tilde{h}$ is then scheduled on a day $\hat{d}$ that maximizes the respective arena-utility difference, $\bar{A}_{\tilde{a},\tilde{h},\hat{d}} - \bar{A}_{a',h',d'}$, where $a'@h'$, currently scheduled on a day d' (which is not necessarily equal to $\hat{d}$), is the conflicting game for the assignment of $\hat{d}$ to $\tilde{a}@\tilde{h}$. The heuristic then reschedules, in a recursive manner, the game $a'@h'$ on another day. Whenever it fails to find a candidate under those conditions, or whenever the maximum recursive level is reached, the free game at hand is scheduled on the corresponding free day leading to the lowest extended incremental cost, determined by f_+, with respect to the current (partial) schedule.

Fleurent's approach-based insertion

Finally, we propose a repair operator that tries to schedule the free games by applying a slightly modified version of the approach by Fleurent (1987). Indeed, the *Fleurent's approach-based insertion* ($\mathsf{05}^+$) is an adaptation of five (out of seven) heuristics addressed in Subsection 5.2.2 on the generation of an initial solution. Although the heuristics are applied in the order suggested by Fleurent (1987), no game exchange is performed by this operator, neither is a forced-home scheduling.

The first modification in the original heuristics refers to the generation of forced trips, but they are still composed of long-distance visits only. In fact, the forced-trip heuristic is split into two phases: at the beginning, it tries to create as many new two-game trips as possible, and only then, it attempts to extend any long-distance trip in the current schedule.

The free-trip heuristic is applied with no modifications. But both the weekend-game and the weekday-game heuristics are changed. They are now applied to all free games, instead of only those referring to short-distance visits, and also, a game may be scheduled within a period for some long-distance trip. In addition, when applying the weekday-game heuristic, which is based on evaluations of "regrets", the cost of assigning a feasible day for a free game now relies only on the free games that are feasible for that day at the time of the evaluation. Finally, because all the assignments are required to be feasible on the Fleurent's heuristics, it may occur that some free games can not be inserted by them into the current schedule. In this case, the same greedy approach as that of the "Max-regret insertion (03^+)" is applied in order to complete the schedule.

The next section reports the computational experiments on our implementation of the ALNS algorithm.

5.3 Computational Experiments

We now describe the experiments conducted when calibrating the ALNS from the preceding section and report some results on several instances of the NHL scheduling problem. The calibration is based on the quality of the solutions that the approach is able to produce. All the solutions are evaluated exclusively by their respective costs, but in some cases, the corresponding number of constraint violations are also reported. For that matter, the weights in the objective function (5.1) were set as follows: $w_1 = 10\,000$, $w_2 = 1\,000$, $w_3 = 10$, $w_4 = 10$, and $w_i = 1$ for each constraint C_i with $i \in \{5, 6, \ldots, 9\}$. Therefore, regarding the actual NHL schedules referred in Table 3.2, the costs for the last five seasons, starting with the 2009-10 season, are 245, 252, 186, 194, and 142, respectively. Obviously, our goal is to design schedules that attain the minimum total cost, which is zero as objective function value, by satisfying all the constraints of the problem.

For all the experiments, the ALNS algorithm was implemented in the object-oriented language COMET™ and run under Linux on a 2.2 GHz Dual AMD Opteron Processor 275 (where only one processor was used) with 8 GB of RAM. By taking advantage of the constraint-based architecture of COMET™ for local search, we were able to quickly evaluate the impact, for each constraint, on the number of violations resulting from every game removal (or insertion). We have implemented such rather high-level abstraction through the definition of invariants and differentiable objects for the NHL scheduling problem, following the notions described in Van Hentenryck and Michel (2009).

The subsections that follows describe the experiments that we have conducted in order to set the main parameters of the ALNS and report some results obtained for several instances

of the problem. In particular, close attention is paid to the choice of a configuration formed by the most promising operators among those proposed in the last section.

In the preceding chapter (p. 50), we have described our generation of instances for the NHL scheduling problem. In particular, some of those outlined in Table 4.2 will now be used for the calibration phase of the ALNS, and then, results on others will be reported at the end of this chapter.

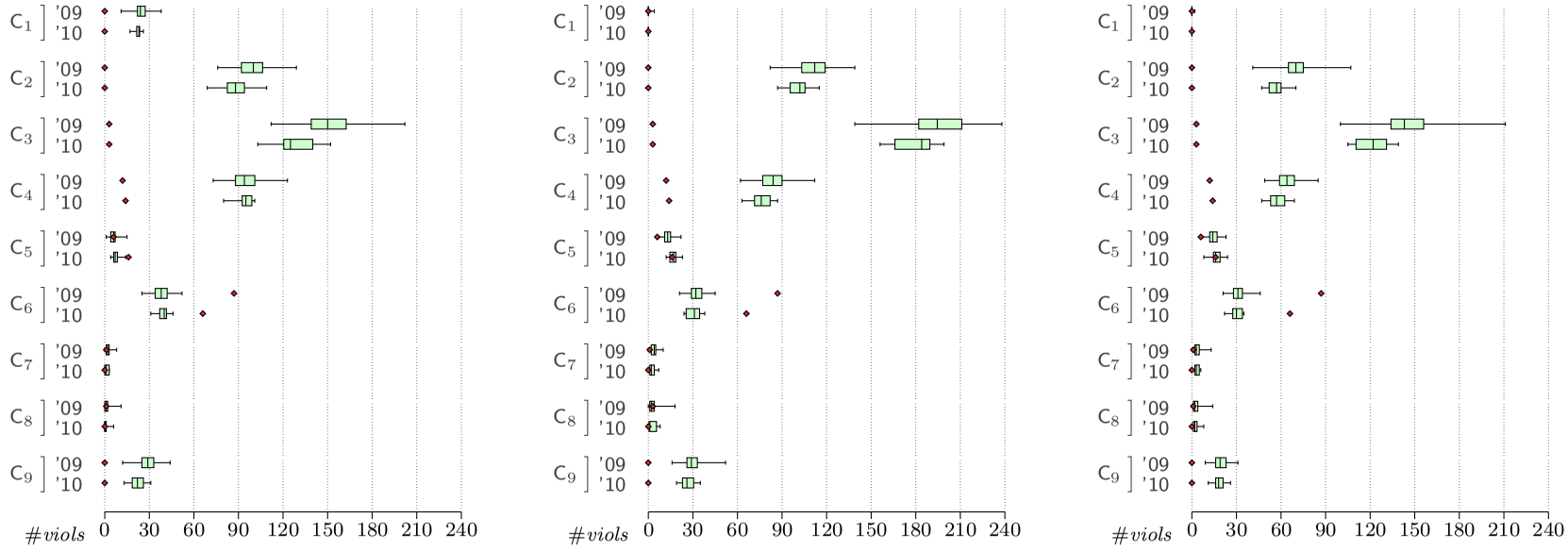
At first, we evaluate the schedules generated by our implementation of the procedures proposed in Fleurent (1987).

5.3.1 Constructing initial solutions

As mentioned earlier, an initial solution must be provided before starting the ALNS iterations. In order to do so in the case of the NHL scheduling problem, we implemented the heuristic procedures first proposed by Fleurent (1987), which are described in Subsection 5.2.2.

Table 5.1 *Statistics on initial schedules* Evaluations on the schedules constructed from scratch by our implementation of Fleurent's heuristics (described in Subsection 5.2.2) for instances of the 2009-10 and 2010-11 regular seasons ('09 and '10, respectively). The multi-heuristic approach was applied 16 times for each one of 21 instances per season: one instance with no extra dates, 10 instances with 4 extra dates per team, and 10 instances with 8 extra dates per team. The table shows the minimum (*min*), the arithmetic mean (*avg*), and the maximum (*max*) costs ($f(\cdot)$) and violations (*#viols*) for each season (Season) and each number of extra arena dates (*NExtraDays*).

	Season	*NExtraDays*	*min*	*avg*	*max*
$f(\cdot)$	'09	0	293 273	348 126.06	398 654
		4	89 338	116 250.95	154 185
		8	42 772	72 165.87	109 986
	'10	0	267 398	313 694.07	357 462
		4	88 255	108 063.46	144 410
		8	38 544	58 483.27	82 367
#viols	'09	0	386	434.13	494
		4	369	471.18	564
		8	270	349.71	474
	'10	0	348	403.47	454
		4	379	446.86	581
		8	226	296.17	395



(a) Instances with no extra dates (b) Instances with 4 extra dates per team (c) Instances with 8 extra dates per team

Figure 5.1 *Constraint violations in initial schedules* Number of constraint violations in the schedules constructed from scratch by our implementation of the heuristic procedures first proposed by Fleurent (1987). The procedure was applied 16 times to each of the 21 selected instances of every season: (a) one instance with no extra dates, (b) 10 instances with 4 extra dates per team, and (c) 10 instances with 8 extra dates per team. The boxplots indicate, for each constraint C_i, $i \in \{1, 2, \ldots, 9\}$, and for each season ('09 and '10), the degree of dispersion in the number of constraint violations on the sample solutions summarized on Table 5.1. The number of violations in the corresponding actual NHL schedule is illustrated by the "diamond" shape ◆.

For any instance created as explained in the preceding subsection, our implementation was able to build from scratch a complete schedule by assigning for each team no more than one game per day, but usually resulting in a large number of constraint violations. The whole construction approach can be seen as a two-stage process where violations for the constraints in C_F are not allowed during the first stage, except perhaps for C_5 (on the minimum number of games per week), and during the second stage, the games that might remain free are sequentially scheduled by now allowing violations for any constraint, except C_0. Thus, the first stage of the process is the same as proposed in Fleurent (1987). In both cases, the heuristic procedures are called exactly in the order they are presented in Subsection 5.2.2 (p. 64), which is in accordance with both Fleurent (1987) and Ferland and Fleurent (1991) for the case of the *batch-mode* (when all games are scheduled without any interaction with the user). Indeed, initially, all free games that individually oppose two distant teams are scheduled at first through the generation of forced trips, which preserves all the arena dates for the respective visiting team, and then, through the generation of free trips, which might sacrifice some of those dates. Subsequently, other free games are scheduled, first to dates that fall on a weekend, and then, to other dates, but never involving any long-distance trip period for the teams in the respective game. And finally, exchange heuristics are used to schedule games that might still remain free. As suggested by Fleurent (1987), the maximum recursive level for the single and for the double exchange heuristics were set to 12 and 3, respectively.

Typical ranges of costs and total number of constraint violations observed for initial solutions are shown in Table 5.1. In addition, Fig. 5.1 illustrates, with a boxplot for each constraint, the degree of dispersion in the number of violations in the sample solutions that had been summarized in Table 5.1. These results, which evaluate initial schedules for the 2009-10 and 2010-11 seasons, refer to 16 runs of the whole construction process for each instance. A total of 21 instances per season, including the one with no extra dates and those with 4 and with 8 extra dates per team, were used in this experiment. In particular, over a sample of 16 runs for each instance with no extra dates, the number of constraint violations and the cost for individual schedules generated were, in all cases, higher than 340 and 250 000, respectively.

When compared to the evaluations for the actual NHL schedules from Table 3.2, these results might be very disappointing. Even for our "easiest" instances, which are made up of 8 extra dates per team, all the generated solutions is by far worse than the actual corresponding schedules. In fact, a high number of the most costly constraints (as C_1, C_2, and C_3) is being violated in these cases.

These results illustrate thus the typical (rather poor) quality of the initial schedule provided to our ALNS. We now evaluate the best-found schedule when varying the number of

games that a destroy operator removes from the current schedule at each iteration of the algorithm.

5.3.2 Choosing the size of the current solution to destroy

Our ALNS algorithm comprises 20 operators, each one having a specific overall goal: 15 of them to partially destroy a complete schedule and the remaining five operators to repair a partial schedule. During each iteration of the algorithm, a destroy operator removes a certain number of games from the current schedule and a repair operator must insert all them back into the schedule. The number of games that a destroy operator is allowed to remove must be provided beforehand and, intuitively, may have a substantial impact on the quality of the solutions produced by the algorithm (as investigated by Pisinger and Ropke, 2007, on variants of the vehicle routing problem, for example).

In this section, we report the experiments we performed on different number of games being removed by the destroy operators. The algorithm was run with all the available operators being uniformly applied through a fixed number of iterations. In other words, at any iteration, every destroy operator had the same selection probability among the destroy operators ($\mathsf{01}^-$, $\mathsf{02}^-$, ..., $\mathsf{15}^-$), and every repair operator had the same selection probability among the repair operators ($\mathsf{01}^+$, $\mathsf{02}^+$, ..., $\mathsf{05}^+$). Hence, the adaptive layer was not used and the approach can be seen as a Large Neighborhood Search (LNS) algorithm, in this case.

Table 5.2 shows some statistics on the best-found solutions generated by the LNS for 11 instances of the 2010-11 regular season. The algorithm was run 10 times for the hardest instance, with no extra dates, and 5 times for every one of the 10 instances with 8 extra dates per team. For each run, an initial solution was generated from scratch. The table lists the minimum (*min*), the arithmetic mean (*avg*), and the maximum (*max*) costs of the best-found solutions for each number of games (*#games*) that was individually removed by the 15 destroy operators through 150 000 LNS iterations with all the available operators being uniformly applied. The last 3 rows of the table refer to a random selection, before each LNS iteration, of an integer value in the interval shown in the first column. On the whole, these experiments comprise several quantities of removed games ranging from 10 to 40.

Unlike the case of the initial solutions from the preceding subsection (see Table 5.1), the evaluations on most of the experimental outcomes in Table 5.2 are somewhat comparable to those of the actual NHL schedule. In particular, when the destroy operators are set to remove 20 games, the costs of the corresponding sample solutions, even for the instance with no extra dates, are lower than the cost (252) of the actual schedule.

Based on these results, the number of games to be removed by any destroy operator was fixed to 20 for all the other experiments we report in this chapter.

Table 5.2 *Results on different number of games being removed per LNS iteration* Evaluations on the best-found solutions generated by our LNS for 11 instances of the 2010-11 regular season. The algorithm was run 10 times for the hardest instance, with no extra dates, and 5 times for every one of the 10 instances with 8 extra dates per team. The table shows the minimum (*min*), the arithmetic mean (*avg*), and the maximum (*max*) costs of the best-found solutions for each number of games (*#games*) removed by any one of the 15 destroy operators through 150 000 LNS iterations with all the available operators being uniformly applied. The last 3 rows of the table refer to a random selection, before each LNS iteration, of an integer value in the interval shown in the first column.

	No extra dates			8 extra dates		
#games	*min*	*avg*	*max*	*min*	*avg*	*max*
10	1 470	7 522.80	21 525	0	0.38	3
15	269	694.80	1 285	0	0.28	2
18	161	234.00	273	0	0.36	2
20	130	**165.75**	214	0	**0.26**	2
25	133	204.00	264	0	0.50	2
30	121	186.60	283	0	0.84	4
40	191	330.25	424	0	2.66	7
16-24	211	238.00	260	0	0.44	2
18-26	199	236.80	299	0	0.44	2
20-28	162	229.00	321	0	0.34	3

The implementation of the ALNS admits several different choices of algorithm control parameters. In the following subsections, we highlight the main computational experiments performed when calibrating the algorithm.

5.3.3 Setting the ALNS parameters

The parameters to be provided to the master level of our ALNS algorithm (see Appendix A.1) are the maximum number of iterations, K_{alns}^{max}; the segment size, K_{seg}; the start temperature control parameter, τ; the score adjustment parameters, σ_{bes}, σ_{bet}, and σ_{acc}; the reaction factor, ρ; the set of destroy operators, I^-, and the set of repair operators, J^+, with their respective initial weights, ω_o $\forall o \in I^- \cup J^+$. There are evidences, both from preliminary results and from the literature, that the values for most of those parameters are generalizable to different instances and, in some cases, even to different problems. Indeed, Ropke and Pisinger (2006), for example, obtained remarkable results for several variants of vehicle routing problems with the same post-tuning values of parameters.

The overall strategy we use here to set the ALNS parameters is the same as that of Ropke and Pisinger (2006). Initially, we have identified an initial setting, and then, one parameter at time was modified while keeping the others fixed, and the modification leading to solutions with the best average cost was chosen for the subsequent parameter adjustments.

The computational experiments reported in the course of this subsection were individually carried out by applying the ALNS eight times to each tuning instance. The cost, and occasionally the number of constraint violations, for the corresponding best-found schedules will be summarized by their minimum (*min*), arithmetic mean (*mean*), and maximum (*max*) values. In most cases, those values will be graphically depicted as *boxplots*, which also includes the lower quartile, the median, and the upper quartile. The setting of parameters will always be based on the average costs of the best-found schedules. So, for each experiment, the setting corresponding to the smallest average cost will be chosen.

Tuning instances and initial solutions

The results in Table 5.1 show that our implementation of the approach by Fleurent (1987) constructs initial solutions that have a rather wide range of values. However, preliminary results that we have obtained while implementing the ALNS (including those from the preceding experiment) showed no relation between the quality of the initial solution and the quality of the best-found solution (even considering just as few as 10 000 iterations). Also during the developing phase, our tests provided compelling evidence that different runs of the ALNS for the same instance and starting at the same initial solution will mostly produce (best-found) schedules with slightly distinct values. We have then decided to select only nine instances for all the experiments in this section. They were, however, carefully chosen as to represent different levels of difficulties. In fact, our choice consists of one instance with no extra dates, four instances with four extra dates per team, and four instances with eight extra dates per team. To be more precise, the tuning instances that we selected are the following: `NHL10-0a`, and for each $\gamma \in \{\mathtt{a}, \mathtt{b}, \mathtt{f}, \mathtt{g}\}$, `NHL10-4`$\gamma$ and `NHL10-8`γ, all them referring to the 2010-11 regular season. In addition, the same initial solution associated with every instance was provided to the experiments. The initial solutions were chosen among those evaluated in Subsection 5.3.1 as to have their respective costs close to the corresponding *mean* value in Table 5.1. Obviously, this choice of solutions is intended to represent a typical initial schedule generated by our implementation. The overall idea is to keep the different tuning steps under the same circumstances regarding both the tuning instances and the initial solutions.

Solution acceptance and stopping criteria

As mentioned in Subsection 5.2.1, our ALNS algorithm iterates until either a schedule S_{curr} with no violation of constraints is generated or a given maximum number of iterations, K_{alns}^{max}, is attained. Because of the strategic nature of the NHL scheduling problem, computational time is not a major issue. So, after a number of *ad hoc* experiments performed during the development phase, we have chosen to set K_{alns}^{max} to 150 000 iterations, which allowed us to obtain all results we report in the remainder of this chapter in no more than 195 minutes (3.25 hours) for each run of the algorithm.

The acceptance criteria we use are those from Simulated Annealing. Therefore, the new (temporary) solution, S_{temp}, obtained during an iteration k of the ALNS is accepted whenever its cost, $f(S_{temp})$, is not worse than the cost, $f(S_{curr})$, of the current solution, S_{curr}; and otherwise, the new solution is accepted with a probability that is exponentially proportional to the quotient of the corresponding increase in the cost, $f(S_{temp}) - f(S_{curr})$, and the current value of the temperature parameter, T_k. As in Ropke and Pisinger (2006), we systematically decrease the temperature at every single iteration of the ALNS. Indeed, the temperature is slightly decreased by using the cooling schedule $T_{k+1} = c\,T_k$ with a constant cooling factor, c, that is close to 1.0. However, instead of following the strategy by Ropke and Pisinger (2006), which provides the cooling factor as a parameter, we set it to a value such that, at the last ALNS iteration, a temporary solution one unit worse than the current solution would have only 1% of chance of being accepted. So we set the constant c to $(T_0\,(-\ln(0.01)))^{(-1/K_{alns}^{max})}$, where T_0 is the initial temperature. This, in turn, is computed as $(\tau/100)(\hat{f}(S_{init})/(-\ln(0.5)))$, where τ is the start temperature control parameter provided to the ALNS and $\hat{f}(S_{init})$ is the weighted sum $\sum_{i=3}^{9} w_i f_i(S_{init})$ of the number of violations, $f_i(S_{init})$, of every constraint C_i, for $i \in \{3, 4, \ldots, 9\}$, in the initial solution, S_{init}. For all the experiments, we have chosen to set $\tau = 5.0$. Therefore, if S_{temp} is the new (temporary) solution at the beginning of the ALNS and S_{temp} induces a cost increasing $f(S_{temp}) - f(S_{init})$ that is equal to 5% of the "modified cost" $\hat{f}(S_{init})$ then S_{temp} is accepted (to become the new S_{curr}) with probability 0.5. The cooling factor in our experiments have always been into the range from 0.999954 to 0.999958, which as expected, results in a slower decrease of the temperature through the ALNS iterations than in the case reported by Ropke and Pisinger (2006), where the cooling rate was set to 0.999750 and the algorithm was run for no more than 25 000 iterations.

We now evaluate the effectiveness of the 20 operators that we proposed in the subsections 5.2.3 and 5.2.4.

Screening the proposed operators

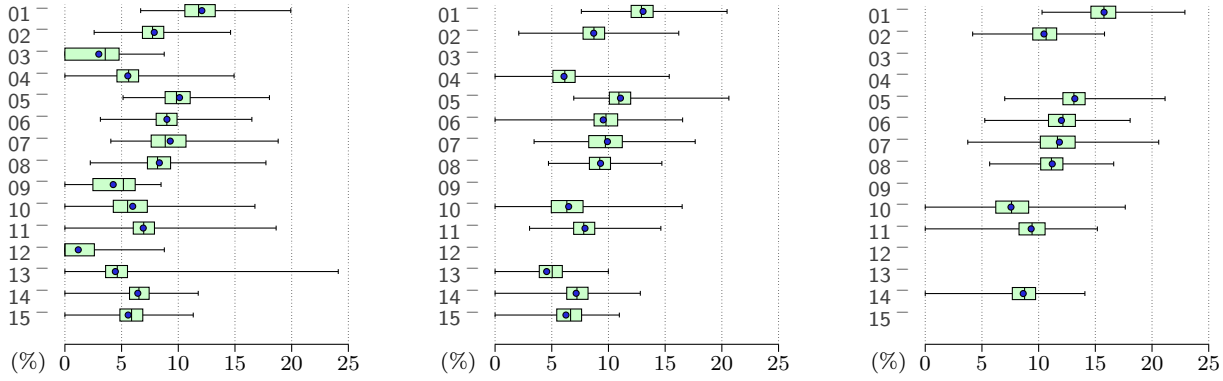
As mentioned earlier, Pisinger and Ropke (2007) state that, when developing an ALNS, the choice between a number of possible operators is not a matter of "either-or" but rather "both-and". Indeed, Ropke and Pisinger (2006) claim that the more (reasonable) operators are provided to their ALNS, the better it performs. However, the literature that follows the same ALNS framework usually reports to apply much fewer operators that we are proposing in this thesis. Our main goal now is, thus, to systematically examine the performance of our ALNS over different configurations for the 20 operators that we have described earlier. In particular, we try to identify a subset of those operators that will, hopefully, outperforms the 20-operator configuration. We also compare the adaptive version of the approach (ALNS) with its static version (LNS), where each operator has a fixed selection probability through the iterations of the search.

For the experiments on the ALNS, we have set the parameters for the adaptive layer as $(K_{seg}, \sigma_{bes}, \sigma_{bet}, \sigma_{acc}, \rho) = (200, 7, 3, 1, 0.1)$, which was based on preliminary results. The values for the score adjustment parameters are then in line with Pisinger and Ropke (2010) as the condition $\sigma_{bes} \geq \sigma_{bet} \geq \sigma_{acc} \geq 0$ holds. This is based upon the rationale that σ_{bes}, σ_{bet}, and σ_{acc} are respectively associated to conditions that are arranged in decreasing order of preference with regard to the quality of the new solution. For example, if it is the case that the solution generated during an ALNS iteration is a (so far) best solution, then the increase σ_{bes} in the scores of the corresponding destroy and repair operators are higher than it would be for any other case.

Ideally, the design of an ALNS algorithm involves both operators that succeed in providing either diversification or intensification through the search. As it is the case in Ropke and Pisinger (2006), the literature on ALNS often proposes operators that individually tends to be very imprecise heuristics, even so, they are able to produce outstanding results when appropriately integrated into an ALNS framework.

Here, we screen our operators in a subtractive manner where, starting from a configuration with all the 20 operators being provided, some apparently less effective operators are excluded through successive computational experiments. Because the number of destroy operators proposed in this thesis is relatively high, we first try to identify (and exclude) more than one operator at time, so as to speed up the process. In fact, by keeping track of the adapted selection probability of each operator through the ALNS iterations, and then by excluding the three lowest-scored operators at once, we were able to reduce the number of destroy operators from 15 to only six operators and yet obtain solutions with better average cost for the tuning instances.

To be more precise, the first phase of this screening process was based on the range



(a) All 15 destroy operators (b) Only 12 destroy operators (c) Only 9 destroy operators

Figure 5.2 *Selection probabilities for each destroy operator through the ALNS iterations* Statistics on the values of the adapted selection probabilities which was tracked at the end of every 10 000 ALNS iterations. All the 5 repair operators are available, and three configurations of destroy operators are individually investigated. In the case (a), all 15 destroy operators are available. In the case (b), only 12 destroy operators are available (operators 03 −, 09 −, and 12 − had been excluded). And in the case (c), only 9 destroy operators are available (operators 04 −, 13 −, and 15 − had also been excluded). For each case, the ALNS was applied 8 times to 9 instances of the 2010-11 regular season: one instance with no extra dates, 4 instances with 4 extra dates per team and other 4 instances with 8 extra dates per team. Thus, the boxplots indicate, for each operator, the degree of dispersion in the selection probabilities tracked at every 10 000 iterations over 72 runs (8 runs × 9 instances) of 150 000 iterations of the ALNS algorithm.

of selection probabilities that are illustrated in the three charts (a, b, and c) of Fig. 5.2 (p. 88). This figure summarizes, in a boxplot for each destroy operator, the range of values of the adapted selection probabilities tracked at the end of every 10 000 ALNS iterations. At the first iteration, in all cases, the same selection probability had been assigned to all the destroy operators available for the respective experiment. The ALNS was run eight times on each tuning instance for at most 150 000 iterations, and during the search, the available operators were individually selected according to their respective earned scores. To be more precise, the Fig. 5.2a refers to the experiment where the initial weight parameters for the operators were set as $\omega_o = 1 \ \forall o \in I^- \cup J^+$, and thus, any destroy operator and any repair operator started with 6.67% and 20% of chance of being selected, respectively. As can be seen from Fig. 5.2a, all selection probabilities turned out to be under 25% for the destroy operators. On the other hand, these results seem to indicate that some destroy operators did not substantially contribute to the generation of good solutions through the search. So, based in Fig. 5.2a, we decided to exclude the operators 03 −, 09 −, and 12 −, which are the three destroy operators with the worst average selection probabilities: 2.99%, 4.26%, and

1.18%, respectively (far below their initial chance of being selected, 6.67%). Fig. 5.2b refers to the experiment with only 12 destroy operators and initial weight parameters set as $\omega_o = 0$ $\forall o \in \tilde{I}_3^- = \{\mathsf{03}^-, \mathsf{09}^-, \mathsf{12}^-\}$, and $\omega_o = 1$ $\forall o \in J^+ \cup I_{12}^-$, with $I_{12}^- = I^- \setminus \tilde{I}_3^-$. As in the preceding case, we inspected the probabilities in Fig. 5.2b and then excluded still the new three worst average-scored operators, which are the operators $\mathsf{04}^-$, $\mathsf{13}^-$, and $\mathsf{15}^-$ (with average selection probabilities of 6.08%, 4.55%, and 6.25%, respectively). Next, we proceeded with the experiment involving only nine destroy operators and the initial weight parameters set as $\omega_o = 0$ $\forall o \in \tilde{I}_6^- = \tilde{I}_3^- \cup \{\mathsf{04}^-, \mathsf{13}^-, \mathsf{15}^-\}$, and $\omega_o = 1$ $\forall o \in J^+ \cup I_9^-$, with $I_9^- = I^- \setminus \tilde{I}_6^-$. For this case, the boxplots on the selection probabilities are given in Fig. 5.2c. One more time, we excluded the three worst average-scored destroy operators, $\mathsf{10}^-$, $\mathsf{11}^-$, and $\mathsf{14}^-$ (with average selection probabilities of 7.56%, 9.34%, and 8.64%, respectively). The experiments on the remaining six destroy operators resulted in a rather more uniform dispersion of the adapted selection probabilities, which does not provide a reasonable evidence for identifying any particularly underperforming operator.

Our criterion on excluding those nine destroy operators, and thus reducing their number from 15 to only six operators, was completely based on the adapted selection probabilities that were sampled through the ALNS iterations. So it is certainly plausible that we now evaluate the quality of the solutions generated by the experiments in each of the preceding cases. We summarize the cost of the best-found solutions for the experiments on 15, 12, 9, and 6 available destroy operators on the top of Fig. 5.3 with the labels I^-, I_{12}^-, I_9^-, and I_6^-, respectively. The results in Fig. 5.3 (p. 98) are arranged, from the top to the bottom, in the order that the experiments were conducted and they are divided into two charts (a and b) each one referring to a particular number of extra dates (0 and 4, respectively) in the tuning instances. The instances with 8 extra dates per team (for which no results are reported in Fig. 5.3) were generally solved to optimality by all the configurations we are describing here. On the whole, the configuration with the 15 destroy operators (I^-) were only fully outperformed (with regard to the average cost) by the configuration with six destroy operators (I_6^-). In particular, the top part of Fig. 5.3 shows that the average cost of the solutions produced by the configuration with 12 operators (I_{12}^-) are the worst among the four preceding experiments. In addition, while the 6-operator configuration (I_6^-) resulted in the lowest average cost on every number of extra dates, the ALNS is apparently more robust for the hardest instance, with no extra dates (Fig. 5.3a), when all the 15 destroy operators are provided to the algorithm.

At this point of our search for a good configuration of the operators, the best average-cost solutions had then been produced by the configuration with six destroy operators, $I_6^- = \{\mathsf{01}^-, \mathsf{02}^-, \mathsf{05}^-, \mathsf{06}^-, \mathsf{07}^-, \mathsf{08}^-\}$, and all the five repair operators, $J^+ = \{\mathsf{01}^+, \mathsf{02}^+, \mathsf{03}^+, \mathsf{04}^+, \mathsf{05}^+\}$.

We aim now to evaluate yet another number of configurations in a subtractive manner that exclude one operator at time and choose the configuration resulting in the best average cost, until no improvement is obtained with this process.

The boxplots labeled by $I_6^- \setminus \{o^-\}$, with every $o^- \in I_6^-$, in Fig. 5.3 illustrate the dispersion in the cost of the best-found solutions when the destroy operator o^- is not provided to the ALNS but only the other five operators in I_6^-. The corresponding results show that the experiment where the operator $\mathsf{07}^-$ was not available was the one to provide solutions with the lowest average cost to the tuning instances. In fact, the configuration with $I_5^- = I_6^- \setminus \{\mathsf{07}^-\}$ provided the (so far) best solutions, compared to the previous experiments. In addition, these results provide compelling evidence that among the operators in I_6^-, the presence of the operators $\mathsf{01}^-$, $\mathsf{02}^-$, and $\mathsf{06}^-$ was particularly important in producing better overall solutions. In the same way, the boxplots labeled by $I_5^- \setminus \{o^-\}$, with every $o^- \in I_5^-$, in Fig. 5.3 summarize the quality of the best-found solutions when the destroy operator o^- was not provided to the ALNS. In this case, despite the rather good overall results on excluding the operator $\mathsf{08}^-$, no average cost in this experiment turned out to be better than the one from our preceding choice, with $I_5^- = \{\mathsf{01}^-, \mathsf{02}^-, \mathsf{05}^-, \mathsf{06}^-, \mathsf{08}^-\}$. Contrary to what might have been expected, this "best" configuration of destroy operators does not include any of the critical removals ($\mathsf{10}^-$,$\mathsf{11}^-$,$\mathsf{12}^-$). In fact, it is rather surprising that a configuration made up of operators among the most naive ones that we have proposed turned out to be the configuration to provide the best average results. Apparently, this could be attributed both to the greedy-based strategy that evaluates the target assignments before cancelling only the "worst" of them and to the relatedness of the target assignments defined by the selection criterion of the other operators (except the random-based removal $\mathsf{01}^-$).

We then evaluated, in the same subtractive manner, the repair operators. As for the destroy operators, they remained unchanged: only the five operators in I_5^- were provided, all them with the same initial weight. The results indicated by a label $J^+ \setminus \{o^+\}$, with every $o^+ \in J^+$, in Fig. 5.3 summarize the costs of the best-found solutions when the repair operator o^+ was not provided to the ALNS but only the other four operators in J^+. As can be seen, a better average cost were obtained by the exclusion of the repair operator $\mathsf{01}^+$. The subsequent experiments, indicated by $J_4^+ \setminus \{o^+\}$, with every $o^+ \in J_4^+$, show that no further single exclusion of repair operators resulted in a more effective configuration than the preceding choice, with $J_4^+ = \{\mathsf{02}^+, \mathsf{03}^+, \mathsf{04}^+, \mathsf{05}^+\}$.

Therefore, based on the strategy we have adopted, our choice of operators for all the experiments that follows is given by $I_5^- = \{\mathsf{01}^-, \mathsf{02}^-, \mathsf{05}^-, \mathsf{06}^-, \mathsf{08}^-\}$ as the set of destroy operators and $J_4^+ = \{\mathsf{02}^+, \mathsf{03}^+, \mathsf{04}^+, \mathsf{05}^+\}$ as the set of repair operators provided to the algorithm.

The three experiments reported at the bottom of Fig. 5.3 refer to the evaluation for non-adaptive versions of the algorithm (LNS), where the selection probabilities were individually fixed through the search, and only the preceding choice of operators, $I_5^- \cup J_4^+$, were provided. Indeed, for the results labeled by *uniform*, the same weight had been assigned to all the operators in $I_5^- \cup J_4^+$, so that, for each iteration, the chance of every destroy operator and every repair operator to be selected was 20% and 25%, respectively. For the other two cases, the setting of selection probabilities was based on statistics for the adapted probabilities from the previous best configuration (labeled by J_4^+ in Fig. 5.3). In fact, in the experiments indicated by *avg10K* and by *avgklast*, the selection probabilities were set as the average of the corresponding adapted probability that had been tracked respectively at all 10 000-iteration blocks and at the last iteration alone. These adapted probabilities are illustrated in Fig. 5.4 both for *all* 10 000-iteration blocks (labeled by A) and for the *last* ALNS iteration alone (labeled by L). We observe from Fig. 5.4 that a quite small dispersion had occurred on the adapted selection probabilities of most operators in the (so far) best configuration. It could then be speculated that the average upon those probabilities for each operator would be the ideal choice for the respective selection probabilities through all the iterations of the algorithm. However, inspection of Fig. 5.3 indicates that neither *avg10K* nor *avgklast* was able to fully outperform the configuration used in J_4^+. Consequently, regarding the average cost of the best-found solutions, these results reinforce the overall superiority of the adaptive version (ALNS) over the static version (LNS) of our algorithm for the NHL scheduling problem.

In all the next experiments, we have then decided to apply the adaptive version of the algorithm (ALNS) with only the operators in $I_5^- \cup J_4^+$ being provided, where in the course of the first iterations (first segment), every destroy operator and every repair operator are selected with 20% and 25% of chance, respectively.

Adjustment of selection probabilities

As described earlier, the selection probabilities for the operators in our ALNS are adjusted though the search according to the following five parameters: the segment size, K_{seg}, that defines the number of ALNS iterations during which scores are added up for each operator; the score adjustment parameters, σ_{bes}, σ_{bet}, and σ_{acc}, that stipulate, at the end of each iteration, the different degrees of "rewards" for the performance of the operators according to the type of new-found solution (*best*, *better*, *accepted*, respectively); and the reaction factor, ρ, that specifies, at the begin of a new segment, how sensitive the updating of any operator weight must be to the corresponding score gathered during the preceding segment. We now report the experiments that we have conducted on evaluating different settings for those parameters.

Ou strategy can be divided into two parts as follows. At first, we kept the same reaction

Table 5.3 *Best-found solutions on different parameters settings for the ALNS* Statistics on the cost of the best-found solution over 150 000 ALNS iterations for three cases of score adjustment parameters, $\sigma = (\sigma_{bes}, \sigma_{bet}, \sigma_{acc}, \sigma_{rej})$, three cases of reaction factor, ρ, and three cases of segment size, K_{seg}. Only 5 destroy operators ($\mathsf{01}^-$, $\mathsf{02}^-$, $\mathsf{05}^-$, $\mathsf{06}^-$, and $\mathsf{08}^-$) and 4 repair operators ($\mathsf{02}^+$, $\mathsf{03}^+$, $\mathsf{04}^+$, and $\mathsf{05}^+$) were provided. The ALNS was applied 8 times to 9 instances of the 2010-11 season, which are investigated by number of extra dates: one instance with no extra dates; 4 instances with 4 extra dates per team; and other 4 instances with 8 extra dates per team. The table shows, for each number of extra dates, the minimum (*min*), the arithmetic mean (*avg*), and the maximum (*max*) best-found values over 72 runs (8 runs × 9 instances) of the ALNS with the corresponding setting of parameters. For each instance, the same initial solution was provided.

ALNS parameters			Instances								
			no extra dates			4 extra dates			8 extra dates		
$(\sigma_{bes}, \sigma_{bet}, \sigma_{acc})$	ρ	K_{seg}	*min*	*avg*	*max*	*min*	*avg*	*max*	*min*	*avg*	*max*
$(33, 9, 13)$	0.1	100	66	116.70	184	0	1.20	4	0	0.10	1
		200	59	104.30	146	0	1.50	6	0	0.05	1
$(33, 13, 9)$	0.1	100	79	96.00	120	0	1.30	3	0	0.05	1
		200	65	91.50	122	0	0.95	3	0	0.00	0
$(7, 3, 1)$	0.1	100	79	99.80	132	0	0.95	3	0	0.10	1
		200	49	89.50	118	0	0.90	2	0	0.00	0
		400	55	93.10	144	0	0.90	3	0	0.05	1
	0.3	100	57	94.50	149	0	1.30	3	0	0.00	0
		200	44	93.60	136	0	0.95	4	0	0.05	1
		400	64	93.10	129	0	0.75	2	0	0.00	0
	0.5	100	63	99.30	139	0	1.25	3	0	0.05	1
		200	54	93.60	134	0	1.15	4	0	0.00	0
		400	61	102.90	147	0	1.15	3	0	0.05	1

factor as before, $\rho = 0.1$, while evaluating all combinations of two cases for the segment size, $K_{seg}^{(1)} = 100$ and $K_{seg}^{(2)} = 200$, and three cases for the vector of score adjustment parameters, $\sigma^{(1)} = (33, 9, 13)$, $\sigma^{(2)} = (33, 13, 9)$, and $\sigma^{(3)} = (7, 3, 1)$. Next, we kept the best among those options for score adjustment parameters, while evaluating all combinations of three cases for the reaction factor, $\rho^{(1)} = 0.1$, $\rho^{(2)} = 0.3$, and $\rho^{(3)} = 0.5$, and also three cases for the segment size, $K_{seg}^{(1)} = 100$, $K_{seg}^{(2)} = 200$, and $K_{seg}^{(3)} = 400$.

The settings $\sigma^{(1)}$, $\rho^{(1)}$, and $K_{seg}^{(1)}$ are those from Ropke and Pisinger (2006), which in particular, somewhat unexpectedly, attribute a higher score increment (13) to "worse" new-

accepted solutions than (9) to new solutions that are "better" with regard to the current solution. Because the setting $\sigma^{(1)}$ is not fully consistent with the condition $\sigma_{bes} \geq \sigma_{bet} \geq \sigma_{acc} \geq 0$ suggested in Pisinger and Ropke (2010) we are then including the setting $\sigma^{(2)}$ into our evalutions. Comparing to our implementation, there is however, one particularity to the adjustments of scores in Ropke and Pisinger (2006): both $\sigma_{bet}^{(1)}$ and $\sigma_{acc}^{(1)}$ are only used for cases where the new solution has not been accepted before. As mentioned earlier, the settings $\sigma^{(3)}$ and $K_{seg}^{(2)}$, which we have used for previous experiments in this thesis, were based in preliminary results obtained during the implementation of our approach. In addition, $\rho^{(2)}$ and $\rho^{(3)}$, as well as $K_{seg}^{(1)}$ and $K_{seg}^{(3)}$, were thought as being reasonable choices to evaluate the impact of a relatively small, medium, and large values for those parameters in the quality of the best-found solutions.

Table 5.3 summarizes the results on those experiments, which are divided by number of extra dates in the respective tuning instances. On the whole, the results does not seem to indicate any particular overall outstanding setting among those we are evaluating. In fact, Table 5.3 seems to suggest that our ALNS for the NHL scheduling problem is not very sensitive to the choice of adjustment parameters. Comparing the results on $\sigma^{(1)}$ and $\sigma^{(2)}$, however, provides some evidence that, contrary to the case in Ropke and Pisinger (2006), attributing decreasing rewards respectively to *best*, *better*, and *accepted* new solutions is quite more effective in our approach for the NHL scheduling problem.

Table 5.3 also shows that, regarding the hardest instance (with no extra dates), our initial setting ($\sigma^{(3)}$, $\rho^{(1)}$, and $K_{seg}^{(2)}$) resulted in the lowest average cost (89.5) and the second lowest overall cost (49) for the best-found solutions. In addition, that setting solved to optimality all tuning instances with 8 extra dates per team.

Therefore, we decided to keep the parameters on the adjustment of selection probabilities as

$$(K_{seg}, \sigma_{bes}, \sigma_{bet}, \sigma_{acc}, \rho) = (200, 7, 3, 1, 0.1)$$

for the results on other instances, which are reported in the following subsection.

5.3.4 Solving other instances

We now report the results obtained by applying the ALNS algorithm to the *entire-league* instances described in Subsection 4.3.2 (p. 50). The parameters on the master level of the algorithm were set as follows: $(\sigma_{bes}, \sigma_{bet}, \sigma_{acc}) = (7, 3, 1)$ for the adjustment of operator scores; $\rho = 0.1$ for the reaction factor; $K_{seg} = 200$ for the segment size; and $K_{alns}^{max} = 150\,000$ for the maximum number of ALNS iterations. Only five destroy operators ($\mathsf{01}^-$, $\mathsf{02}^-$, $\mathsf{05}^-$, $\mathsf{06}^-$, and $\mathsf{08}^-$) and four repair operators ($\mathsf{02}^+$, $\mathsf{03}^+$, $\mathsf{04}^+$, and $\mathsf{05}^+$) were provided, and an initial schedule

Table 5.4 *Best-found solutions for instances with extra arena-available dates* Statistics on the cost of the best-found solution over 150 000 ALNS iterations when the initial solution is built from scratch. The results refer to 10 instances (a, b, . . . , j) on two and on four extra arena-available dates per team for the 2010-11 season. The table shows, for each instance, the minimum (*min*), the arithmetic mean (*avg*), and the maximum (*max*) best-found costs over 8 runs of the ALNS algorithm. Instances with either 6 or 8 extra arena dates per team were all solved to optimality (with no constraint violations).

		2 extra dates			4 extra dates		
α	γ	*min*	*avg*	*max*	*min*	*avg*	*max*
2 010	a	1	1.25	3	0	0.38	1
	b	1	1.75	2	0	0.38	1
	c	1	2.75	4	0	0.38	1
	d	0	1.50	3	0	0.25	1
	e	1	2.00	3	0	0.13	1
	f	0	2.88	4	0	0.63	1
	g	1	3.00	5	0	1.00	2
	h	0	1.38	4	0	0.63	2
	i	0	2.63	6	0	0.63	1
	j	2	3.00	4	0	0.25	1
	all	0	2.21	6	0	0.46	2

was generated from scratch for each run of the algorithm.

As mentioned in Chapter 3, the arena availability is by far the most critical factor in the construction of schedules for the NHL regular season. It is then logical to assume that, in particular, the less extra dates an instance has, the more difficult it is to find a high-quality schedule for it. This would suggest that even our "easiest" instances (with 8 extra dates per team) are substantially more difficult to solve than those addressed either by Fleurent (1987) or by Costa (1995), where the instances are reported to have at least 15 extra arena dates per team. On the other hand, because no further information about the instances addressed in those works is available, we believe that any attempt to rigorously compare those instances with ours would be rather misleading. In fact, it may happen that the difficulty of our instances are somewhat attenuated by the fact of having been generated from an already-known fairly good solution: the corresponding actual NHL schedule.

Tables 5.4 and 5.5 present some statistics on the cost of the best-found solution in our computational experiments for the 2010-11 season, which turns out to fairly represent the typical solutions our ALNS have provided on the entire-league instances.

Table 5.5 *Typical results for instances without extra dates* Statistics on the constraint violations and on the costs of the best-found solution over 150 000 ALNS iterations when the initial solution is built from scratch. The results refer to the instance `NHL10-00a` of the 2010-11 regular season with no extra arena-available dates. The table shows the minimum (*min*), the arithmetic mean (*avg*), and the maximum (*max*) number of constraint violations (*ctr*) and costs ($f(\cdot)$) on the best-found solutions over 8 runs of the ALNS algorithm. Constraint violations and cost are also show for the corresponding actual NHL schedule (*NHL'*).

α	*ctr*	*min*	*avg*	*max*	*NHL'*
2 010	C_1	0	0.00	0	0
	C_2	0	0.00	0	0
	C_3	0	0.13	1	3
	C_4	4	7.38	10	14
	C_5	0	0.13	1	16
	C_6	6	10.00	19	66
	C_7	0	0.00	0	0
	C_8	0	0.00	0	0
	C_9	0	0.25	1	0
	#viols	10	17.88	27	99
	$f(\cdot)$	46	85.38	120	252

The results are divided into two parts as follows: Table 5.4 refers to the instances with extra arena dates, and Table 5.5 refers to the instances where only the 41 home dates in the corresponding actual NHL schedule were provided. In particular, Table 5.5 also presents the number of violations for each constraint of the problem, regarding both the solutions provided by the ALNS algorithm and the respective actual NHL schedules.

On the whole, the results reinforce the importance of having extra arena dates in order for our ALNS to find, within the considered number of iterations, NHL schedules that violate no constraints of the problem. Indeed, all the instances with either 6 or 8 extra arena dates per team (which it is not shown in the tables) were solved to optimality (with no constraint violations). Moreover, the statistics on this sample of eight schedules per instance for the 2010-11 season show that the numbers of violations are on average only 2.21 and 0.46 for the cases where each team provided 2 and 4 extra arena dates, respectively.

It is important to notice that the violations reported in Table 5.4 can only refer to the "cheapest" constraints, to which we have set a one-unit penalty weight (or cost) per violation.

When only the minimum number of arena dates (41) is provided by each team, table 5.5 shows, however, that the number of violations in the schedules generated by the ALNS

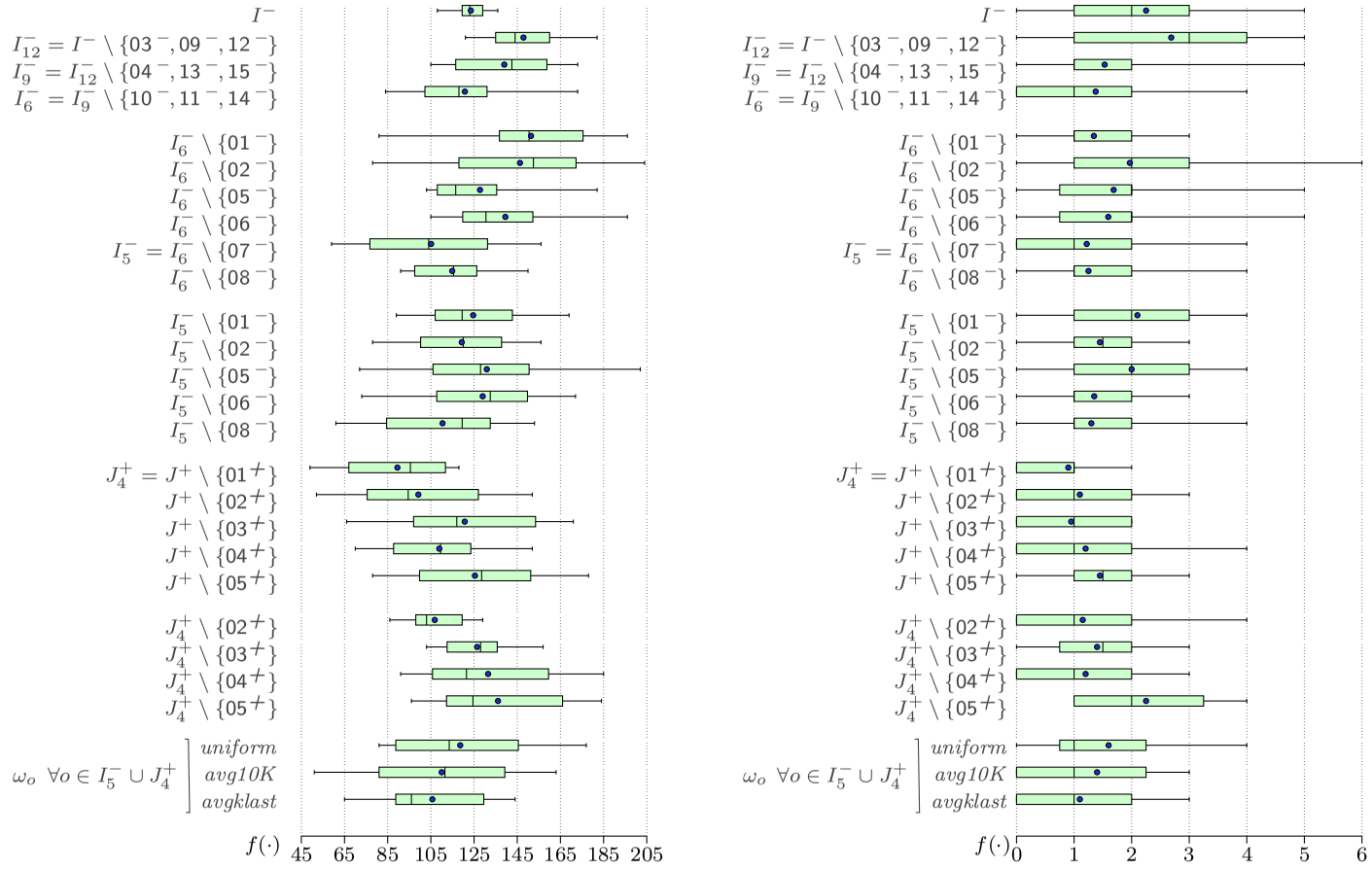
algorithm ranges from 10 to 27, with cost varying between 46 and 120. Although these violations and costs are worse (higher) than those for the other instances (Table 5.4), they are significantly better (lower) than the violations (99) and the cost (252) for the actual 2010-11 NHL schedule. Interestingly, the algorithm was thus able to reschedule the games on the same arena dates that had actually been used by every team and yet violates much fewer constraints of the problem.

As shown by Table 5.6, similar comparative remarks could be drawn about the results for the instances of other seasons. In particular, however, some schedules have presented slightly more violations than those for the 2010-11 season. For example, we have noticed that slightly worse schedules were generated for the 2009-10 season. But that was not completely unexpected, and it could be attributed to the fact that fewer playable dates were available for this particular season, as the XXI Winter Olympic Games had shut out nearly two weeks from the regular-season schedule in January of 2010.

On the whole, our results seem to be significantly better than those reported in Costa (1995). In particular, with regard to the 1993-94 season, when the NHL was formed by only 24 teams but every team already had to play 41 home games, Costa's best result refer to a schedule that, even though at least 15 extra dates had been provided by each team, exactly 105 games were scheduled on days when the arena of the respective home team was not available. In addition, that schedule causes two violations for the constraints C_2 and C_3, and it also presents 33 occurrences of a team having byes made up of more than three consecutive days in the course of a single road trip. With respect to our best schedules, even for the instances with no extra arena dates, none of those requirements were violated.

Table 5.6 *Best-found schedules for instances without extra arena-available dates* Constraint violations on the best-found solutions by the ALNS for the entire-league instances without extra arena-available dates, and violations on the corresponding official NHL schedules.

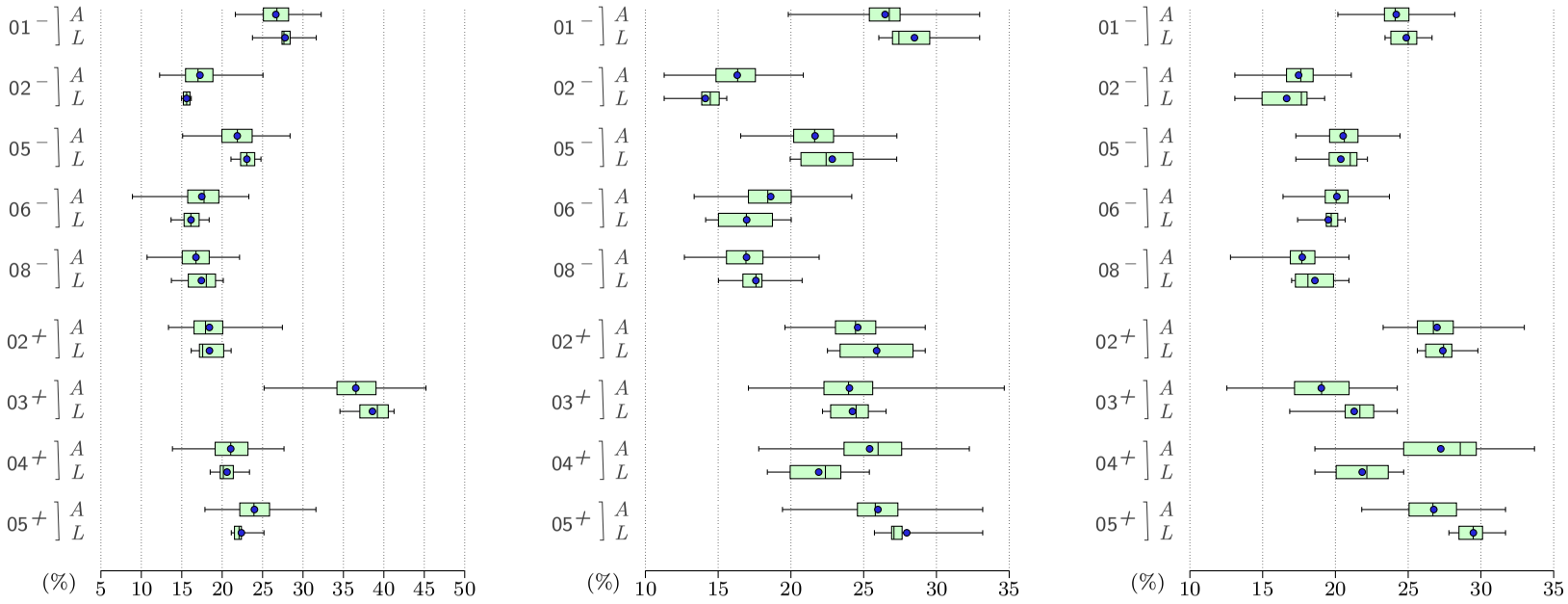
		Number of violations								
Season	By	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
'09	ALNS	0	0	0	10	3	12	0	0	0
	NHL	0	0	3	12	6	87	1	1	0
'10	ALNS	0	0	0	4	0	6	0	0	0
	NHL	0	0	3	14	16	66	0	0	0
'11	ALNS	0	0	0	5	4	8	0	0	0
	NHL	0	0	0	9	19	75	1	1	0
'12	ALNS	0	0	0	9	1	21	0	0	0
	NHL	0	0	0	10	3	88	1	1	0
'13	ALNS	0	0	0	0	3	8	0	0	0
	NHL	0	0	0	10	8	33	1	1	0



(a) Instance with no extra dates

(b) Instances with 4 extra dates per team

Figure 5.3 *Best-found solutions on different configurations of operators* Statistics on the costs of the best-found solutions over 150 000 (A)LNS iterations. The computational experiments were conducted in the order they appear, from the top to the bottom, both for the (a) instance with no extra dates and for the (b) instances with 4 extra dates. At the beginning (label I^-), all the 20 operators, $I^- \cup J^+$, were provided, but operators have been successively excluded (as indicated by the respective set-theoretic difference) until only nine operators remained, which are $I^-_5 = \{01^-, 02^-, 05^-, 06^-, 08^-\}$ and $J^+_4 = \{02^+, 03^+, 04^+, 05^+\}$. All results refer to the adaptive version of the algorithm (ALNS), except for the last three experiments (in the bottom of each chart). For each experiment, the algorithm was applied 8 times to five instances of the 2010-11 season: one instance with no extra dates and four instances with 4 extra dates per team. Thus, the boxplots indicate, for each configuration of operators, the degree of dispersion in the costs of the best-found solutions over 40 runs (8 runs $\times$ 5 instances) of 150 000 iterations for the (A)LNS.



(a) Instance with no extra dates (b) Instances with 4 extra dates per team (c) Instances with 8 extra dates per team

Figure 5.4 *Selection probabilities for the chosen operators through the ALNS iterations* Statistics on the values of the adapted selection probabilities tracked at the end of every 10 000 ALNS iterations (A) and, in particular, at the last ALNS iteration alone (L). Only 5 destroy operators ($\mathsf{01}^-$, $\mathsf{02}^-$, $\mathsf{05}^-$, $\mathsf{06}^-$, and $\mathsf{08}^-$) and 4 repair operators ($\mathsf{02}^+$, $\mathsf{03}^+$, $\mathsf{04}^+$, and $\mathsf{05}^+$) were provided. The ALNS was applied 8 times to 9 instances of the 2010-11 season, which are individually investigated: (a) one instance with no extra dates, (b) 4 instances with 4 extra dates per team, and (c) other 4 instances with 8 extra dates per team. Thus, the boxplots indicate, for each operator, the degree of dispersion in the selection probabilities tracked over 72 runs (8 runs $\times$ 9 instances) of the ALNS algorithm.

CHAPTER 6

CONCLUSION

In this thesis, we have revisited the NHL scheduling problem, which consists in scheduling the regular-season games of the National Hockey League. Some basic requirements for a good NHL schedule were identified, and then, we proposed the first integer linear programming (IP) model for exactly solving the problem and also an adaptive large neighborhood search (ALNS) for solving instances of practical size. In particular, we have reported some computational results for instances of moderate size that were obtained from a commercial state-of-the-art solver with several variants of the IP model, where different constraints have been relaxed and their violations penalized in the objective function. Promising results for instances of realistic size that were generated by the ALNS have also been reported. Indeed, the ALNS was able to find schedules that satisfy all essential requirements even for the most challenging instances of the problem where, in particular, the availability of arena is scarce or at its minimum. Such results reinforce the original idea that high-quality solutions for tightly-constrained problems can be achieved by a combination of rather naive (and often individually ineffective) heuristics into an ALNS framework.

As we have mentioned in the literature review, the focus of studies on sports scheduling has often been the minimization of travel distances. In such cases, distances are somewhat used as measure of travelling costs and players fatigue. This can be controversial as most travels are done by air, and then, in particular, fatigue would be more due to poor quality of the travel experience than long distance itself. Moreover, some particular game scheduling may result in a slightly longer travelling distance for the teams involved but, for example, have the advantage of the game being scheduled during a weekend, which could increase spectator attendance or broadcasting revenue, and thus fairly compensate some possible "extra" travel cost. And also, because the home locations of the NHL teams are so unevenly distributed throughout a vast region, care should be taken as not only try to minimize the overall travel distance at the expense of aggravating the particular situation of the most geographically isolated teams. Therefore, although we recognize that travels are indeed one of the concerns in the NHL scheduling problem, in the case of the ALNS, we have considered them only implicitly by the use of the saving measure in the generation of both forced and free trips.

Our ALNS algorithm for the NHL scheduling problem is well suited for the cases where some games have been scheduled beforehand. In fact, all the results that we have reported in this thesis take into account that, in particular, the NHL regular season currently includes

a few games played in Europe, and also other special games, at predetermined dates that cannot be changed.

We are confident that by involving the expert scheduler into a detailed formulation of the problem and into the implementation of other specialized heuristics in the ALNS framework would enable computers to assist the designing of much better schedules than the NHL might have conceived as being possible. In fact, the reported results suggest that our approach could enable the NHL managers to identify unnecessary weakness in their schedules, and could certainly assist them in finding high-quality schedules with respect to a variety of new preferences.

Starting in the 2013-14 season, the NHL adopted a new structure, which distributes the 30 teams into only two eight-team divisions and two seven-team divisions. Furthermore, the teams are now playing under a new regular-season structure, yet a team still plays 41 home games. It would be useful to investigate the impact of such changes in the quality of the schedules that our approach is able to produce with regard to the essential requirements we have considered in this thesis and possibly other preferences and measures on fairness issues for the teams.

REFERENCES

ARMSTRONG, J. and WILLIS, R. J. (1993). Scheduling the Cricket World Cup – a case study. *The Journal of the Operational Research Society*, 44, 1067–1072.

BAO, R. (2009). *Time Relaxed Round Robin Tournament and the NBA Scheduling Problem*. Ph.D. thesis, Mechanical Engineering, Cleveland State University. Cleveland, United States.

BARTSCH, T., DREXL, A. and KRÖGER, S. (2006). Scheduling the professional soccer leagues of Austria and Germany. *Computer & Operations Research*, 33, 1907–1937.

BEAN, J. C. and BIRGE, J. R. (1980). Reducing Travelling Costs and Player Fatigue in the National Basketball Association. *Interfaces*, 10, 98–102.

BRISKORN, D. (2008a). Feasibility of home-away-pattern sets for round robin tournaments. *Operations Research Letters*, 36, 283–284.

BRISKORN, D. (2008b). *Sports Leagues Scheduling: Models, Combinatorial Properties, and Optimization Algorithms*. Lecture Notes in Economics and Mathematical Systems, 603. Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg.

BRISKORN, D. and DREXL, A. (2007). A branch-and-price algorithm for scheduling sport leagues. *Journal of the Operations Research Society*, 60, 89–93.

BROUWER, A. E., POST, G. F. and WOEGINGER, G. J. (2008). Tight bounds for break minimization in tournament scheduling. *Journal of Combinatorial Theory, Series A*, 115, 1065–1068.

CHEUNG, K. K. H. (2008). Solving mirrored traveling tournament problem benchmark instances with eight teams. *Discrete Optimization*, 5, 138–143.

CLARKE, G. and WRIGHT, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12, 568–581.

COSTA, D. (1995). An evolutionary tabu search algorithm and the NHL scheduling problem. *INFOR*, 33, 161–178.

CRAIG, S., WHILE, L. and BARONE, L. (2009). Scheduling for the National Hockey League Using a Multi-objective Evolutionary Algorithm. In A. Nicholson and X. Li, editors, *AI 2009: Advances in Artificial Intelligence*, Springer Berlin Heidelberg, vol. 5866 of *Lecture Notes in Computer Science*, book section 39. Pages 381–390.

DE WERRA, D. (1980). Geography, games and graphs. *Discrete Applied Mathematics*, 2, 327–337.

DE WERRA, D. (1981). Scheduling in Sports. In P. Hansen, editor, *North-Holland Mathematics Studies*, North-Holland, vol. 59. Pages 381–395.

DE WERRA, D. (1982). Minimizing irregularities in sports schedules using graph theory. *Discrete Applied Mathematics*, 4, 217–226.

DE WERRA, D. (1985). On the multiplication of divisions: the use of graphs for sports scheduling. *Networks*, 15, 125–136.

DE WERRA, D. (1988). Some models of graphs for scheduling sports competitions. *Discrete Applied Mathematics*, 21, 47–65.

DELLA CROCE, F. and OLIVERI, D. (2006). Scheduling the Italian Football League: an ILP-based approach. *Computers & Operations Research*, 33, 1963–1974.

DEPALMA, T. J. (2004). *Betting market efficiency in the National Hockey League: An analysis of expansion seasons*. Ph.D. thesis, Faculty of Economics, University of Delaware. Delaware, United States.

DREXL, A. and KNUST, S. (2007). Sports league scheduling: graph- and resource-based models. *Omega*, 35, 465–471.

EASTON, K., NEMHAUSER, G. and TRICK, M. (2001). The Traveling Tournament Problem Description and Benchmarks. In T. Walsh, editor, *Principles and Practice of Constraint Programming — CP 2001*, Springer Berlin Heidelberg, vol. 2239 of *Lecture Notes in Computer Science*, book section 43. Pages 580–584.

EASTON, K., NEMHAUSER, G. and TRICK, M. (2003). Solving the traveling tournament problem: a combined integer programming and constraint programming approach. In E. Burke and P. De Causmaecker, editors, *Practice and Theory of Automated Timetabling IV (PATAT 2002)*. Spring-Verlag, vol. 2740. Pages 100–109.

EASTON, K., NEMHAUSER, G. and TRICK, M. (2004). Sport Scheduling. In J. Y.-T. Leung, editor, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, CRC Press, book section 52. Pages 1139–1157.

EASTON, K. K. (2003). *Using integer programming and constraint programming to solve sports scheduling problems*. Ph.D. thesis, Georgia Institute of Technology, Georgia, United States.

EASTON, T. and GARY PARKER, R. (2001). On completing latin squares. *Discrete Applied Mathematics*, 113, 167–181.

ELF, M., JÜNGER, M. and RINALDI, G. (2003). Minimizing breaks by maximizing cuts. *Operations Research Letters*, 31, 343–349.

FERLAND, J. A. and FLEURENT, C. (1991). Computer aided scheduling for a sport league. *INFOR*, 29, 14–25.

FLEURENT, C. (1987). *Méthodes heuristiques pour la conception de calendriers sportifs*. Master's thesis, Département d'informatique et de recherche opérationnelle, Université de Montréal. Montreal, Quebec.

FLEURENT, C. and FERLAND, J. A. (1993). Allocating games for the NHL using Integer Programming. *Operations Research*, 41, 649–654.

FRASER, W. (1982). *The role of computer simulation in building the National Hockey League Schedule*. Technical report, IBM Canada Limited.

GOEMANS, M. X. and WILLIAMSON, D. P. (1995). Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming. *J. ACM*, 42, 1115–1145.

GOOSSENS, D. and SPIEKSMA, F. (2011). Soccer schedules in Europe: an overview. *Journal of Scheduling*, 15, 651–651.

HAMIEZ, J.-P. and HAO, J.-K. (2004). A linear-time algorithm to solve the Sports League Scheduling Problem (prob026 of CSPLib). *Discrete Applied Mathematics*, 143, 252–265.

HAVARD, C. T. (2014). Glory Out of Reflected Failure: The examination of how rivalry affects sport fans. *Sport Management Review*, 17, 243–253.

HENZ, M. (2001). Scheduling a Major College Basketball Conference–Revisited. *Operations Research*, 49, 163–168.

IRNICH, S. (2010). A new branch-and-price algorithm for the traveling tournament problem. *European Journal of Operational Research*, 204, 218–228.

KENDALL, G., KNUST, S., RIBEIRO, C. C. and URRUTIA, S. (2010). Scheduling in sports: An annotated bibliography. *Computers & Operations Research*, 37, 1–19.

KNUST, S. (2010). Scheduling non-professional table-tennis leagues. *European Journal of Operational Research*, 200, 358–367.

KNUST, S. (2014). *Classification of Literature on Sports Scheduling*. Webpage `http://www.inf.uos.de/knust/sportlit_class/sportlit_class.html`. Last checked on May 10, 2014.

KNUST, S. and LÜCKING, D. (2009). Minimizing costs in round robin tournaments with place constraints. *Computers & Operations Research*, 36, 2937–2943.

KNUST, S. and VON THADEN, M. (2006). Balanced home-away assignments. *Discrete Optimization*, 3, 354–365.

KOVACS, A. A., PARRAGH, S. N., DOERNER, K. F. and HARTL, R. F. (2012). Adaptive large neighborhood search for service technician routing and scheduling problems. *Journal of Scheduling*, 15, 579–600.

LAMKEN, E. R. (1990). Generalized Balanced Tournament Designs. *Transactions of the American Mathematical Society*, 318, 473–490.

LARSON, J. and JOHANSSON, M. (2014). Constructing Schedules for Sports Leagues with Divisional and Round-robin Tournaments. *Journal of Quantitative Analysis in Sports*, 10, 119–129. doi:10.1515/jqas-2013-0090.

LENTEN, L. J. A. (2013). Measurement of Competitive Balance in Conference and Divisional Tournament Design. *Journal of Sports Economics*, 1–23. doi:10.1177/1527002512471538.

LIM, A., RODRIGUES, B. and ZHANG, X. (2006). A simulated annealing and hill-climbing algorithm for the traveling tournament problem. *European Journal of Operational Research*, 174, 1459–1478.

MACDONALD, B. and PULLEYBLANK, W. (2014). Realignment in the NHL, MLB, NFL, and NBA. *Journal of Quantitative Analysis in Sports*, 10, 225–240. doi:10.1515/jqas-2013-0070.

MELO, R. A., URRUTIA, S. and RIBEIRO, C. C. (2007). Scheduling Single Round Robin Tournaments with Fixed Venues. In *3rd Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA'07)*. Pages 431–438.

MIYASHIRO, R., IWASAKI, H. and MATSUI, T. (2003). Characterizing feasible pattern sets with a minimum number of breaks. In E. Burke and P. De Causmaecker, editors, *Practice and Theory of AutomatedTimetabling IV*, Springer-Verlag, Berlin Heidelberg, LNCS 2740. Pages 78–99.

MIYASHIRO, R. and MATSUI, T. (2005). A polynomial-time algorithm to find an equitable home-away assignment. *Operations Research Letters*, 33, 235–241.

MIYASHIRO, R. and MATSUI, T. (2006). Semidefinite programming based approaches to the break minimization problem. *Computers & Operations Research*, 33, 1975–1982.

NEMHAUSER, G. and TRICK, M. (1998). Scheduling a Major College Basketball Conference. *Operations Research*, 46, 1–8.

NURMI, K., GOOSSENS, D., BARTSCH, T., BONOMO, F., BRISKORN, D., DURAN, G., KYNGÄS, J., MARENCO, J., RIBEIRO, C., SPIEKSMA, F. *ET AL.* (2010). A framework for a highly constrained sports scheduling problem. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*. vol. 3. Pages 1991–1997.

OBERHOFER, H., PHILIPPOVICH, T. and WINNER, H. (2010). Distance matters in away games: Evidence from the German football league. *Journal of Economic Psychology*, 31, 200–211.

PISINGER, D. and ROPKE, S. (2007). A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34, 2403–2435.

PISINGER, D. and ROPKE, S. (2010). Large Neighborhood Search. In M. Gendreau and J.-Y. Potvin, editors, *Handbook of Metaheuristics*, Springer US, vol. 146 of *International Series in Operations Research & Management Science*. Pages 399–419.

POST, G. and WOEGINGER, G. J. (2006). Sports tournaments, home-away assignments, and the break minimization problem. *Discrete Optimization*, 3, 165–173.

RASMUSSEN, R. and TRICK, M. (2006). The Timetable Constrained Distance Minimization Problem. In J. C. Beck and B. Smith, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Springer Berlin Heidelberg, vol. 3990 of *Lecture Notes in Computer Science*, book section 15. Pages 167–181.

RASMUSSEN, R. and TRICK, M. (2009). The timetable constrained distance minimization problem. *Annals of Operations Research*, 171, 45–59.

RASMUSSEN, R. V. (2008). Scheduling a triple round robin tournament for the best Danish soccer league. *European Journal of Operational Research*, 185, 795–810.

RASMUSSEN, R. V. and TRICK, M. A. (2007). A Benders approach for the constrained minimum break problem. *European Journal of Operational Research*, 177, 198–213.

RASMUSSEN, R. V. and TRICK, M. A. (2008). Round robin scheduling – a survey. *European Journal of Operational Research*, 188, 617–636.

RÉGIN, J.-C. (2001). Minimization of the number of breaks in sports scheduling problems using constraint programming. In E. C. Freuder and R. J. Wallace, editors, *Constraint programming and large scale discrete optimization*, American Mathematical Society, Rutgers, New Jersey, vol. 57 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. Pages 115–130.

REINFELD, N. V. and VOGEL, W. R. (1958). *Mathematical programming*. Englewood Cliffs, N. J. : Prentice-Hall.

RIBEIRO, C. C. (2012). Sports scheduling: Problems and applications. *International Transactions in Operational Research*, 19, 201–226.

ROKOSZ, F. M. (2000). *Procedures for structuring and scheduling sports tournaments: elimination, consolation, placement, and round-robin design*. C.C. Thomas, Springfield, Ill., third edition.

ROPKE, S. and PISINGER, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40, 455–472.

ROSEN, D. (2013). *Realignment plan approved by Board of Governors*. Webpage `http://www.nhl.com/ice/news.htm?id=660138`. Last checked on July 3, 2014.

RUSSELL, R. A. and LEUNG, J. M. Y. (1994). Devising a cost effective schedule for a baseball league. *Operations Research*, 42, 614–625.

SCHAERF, A. (1999). Scheduling sport tournaments using constraint logic programming. *Constraints*, 4, 43–65.

SCHÖNBERGER, J., MATTFELD, D. C. and KOPFER, H. (2000). Automated timetable generation for rounds of a table-tennis league. In *Evolutionary Computation, 2000 – Proceedings of CEC2000*. vol. 1. Pages 277–284.

SCHÖNBERGER, J., MATTFELD, D. C. and KOPFER, H. (2004). Memetic Algorithm timetabling for non-commercial sport leagues. *European Journal of Operational Research*, 153, 102–116.

SCHREUDER, J. (1992). Combinatorial aspects of construction of competition Dutch professional football leagues. *Discrete Applied Mathematics*, 35, 301–312.

SCHREUDER, J. A. M. (1980). Constructing timetables for sport competitions. *Mathematical Programming Study*, 13, 58–67.

SCHRIMPF, G., SCHNEIDER, J., STAMM-WILBRANDT, H. and DUECK, G. (2000). Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159, 139–171.

SHAW, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. *Principles and Practice of Constraint Programming - Cp98*, 1520, 417–431.

SMITH, D. R., CIACCIARELLI, A., SERZAN, J. and LAMBERT, D. (2000). Travel and the home advantage in professional sports. / Avantage du terrain par rapport a un deplacement pour les sportifs professionnels. *Sociology of Sport Journal*, 17, 364–385.

SUZUKA, A., MIYASHIRO, R., YOSHISE, A. and MATSUI, T. (2007). The home-away assignment problems and break minimization/maximization problems in sports scheduling. *Pacific Journal of Optimization*, 3, 113–133.

TRICK, M. (2001). A schedule-then-break approach to sports timetabling. In E. Burke and W. Erben, editors, *Practice and Theory of Automated Timetabling III*, Springer Berlin Heidelberg, vol. 2079 of *Lecture Notes in Computer Science*. Pages 242–253.

TRICK, M. (2005). Formulations and Reformulations in Integer Programming. In R. Barták and M. Milano, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Springer Berlin Heidelberg, vol. 3524 of *Lecture Notes in Computer Science*, chapter 27. Pages 366–379.

TRICK, M. (2011). Sports Scheduling. In P. van Hentenryck and M. Milano, editors, *Hybrid Optimization*, Springer New York, vol. 45 of *Springer Optimization and Its Applications*. Pages 489–508.

URRUTIA, S. and RIBEIRO, C. C. (2006). Maximizing breaks and bounding solutions to the mirrored traveling tournament problem. *Discrete Applied Mathematics*, 154, 1932–1938.

UTHUS, D., RIDDLE, P. and GUESGEN, H. (2009). DFS* and the Traveling Tournament Problem. In W.-J. Hoeve, J. Hooker, D. Uthus, P. Riddle and H. Guesgen, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Springer Berlin Heidelberg, vol. 5547 of *Lecture Notes in Computer Science*. Pages 279–293.

UTHUS, D., RIDDLE, P. and GUESGEN, H. (2012). Solving the traveling tournament problem with iterative-deepening A*. *Journal of Scheduling*, 15, 601–614.

VAN HENTENRYCK, P. and MICHEL, L. (2009). *Constraint-Based Local Search*. The MIT Press.

VAN HENTENRYCK, P. and VERGADOS, Y. (2005). Minimizing breaks in sport scheduling with local search. In *Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS) 2005*. vol. 1. Pages 22–29. June 5-10, 2005, Monterey, California, United States.

VAN WEERT, A. and SCHREUDER, J. (1998). Construction of basic match schedules for sports competitions by using graph theory. In E. Burke and M. Carter, editors, *Practice and Theory of Automated Timetabling II*, Springer Berlin Heidelberg, vol. 1408 of *Lecture Notes in Computer Science*, book section 13. Pages 201–210.

APPENDIX A

PSEUDOCODE OF THE ALNS ALGORITHM

A.1 Pseudocode of the ALNS algorithm

The main features of the ALNS algorithm we propose for the NHL Scheduling Problem can be summarized as stated in Algorithm 1. It essentially consists of the construction of an initial solution (S_{init}) and a loop that, while not all constraints are satisfied (or equivalently while $f(S_{best}) > 0$), tries to improve the current solution (S_{curr}) through a fixed number of iterations (with k varying from 0 to $K_{alns}^{max} - 1$). The parameters provided to the algorithm are the maximum number of iterations, K_{alns}^{max}; the segment size, K_{seg}; the start temperature control parameter, τ; the score adjustment parameters, σ_{bes}, σ_{bet}, and σ_{acc}; the reaction factor, ρ; the set of destroy operators, I^-, and the set of repair operators, J^+, with their respective initial weights, ω_o $\forall o \in I^- \cup J^+$. In order for us still to focus on the master level of the approach, the statement of the algorithm evokes three "external" procedures (denoted with a `typewriter` style font), which we describe in a rather informal way. On line 1, a complete schedule is constructed as the initial solution, S_{init}, by the procedure `constructInitialSolution()`. This procedure implements the multi-heuristic approach proposed by Fleurent (1987), which we describe in the next subsection. In the body of the loop (from lines 9 to 33), one destroy operator and one repair operator are initially selected by two calls, on lines 9 and 10, to the procedure `selectOperator(·)`, whose argument can be seen as a vector of weights for the random weighted selection among the corresponding candidate operators. That procedure implements thus the strategy on the operator selection explained earlier. The just-selected pair of destroy and repair operators are applied to the current solution by the procedure `applyOperators(·)` on line 11. The resulting destroyed-and-repaired solution is then either accepted or rejected by the procedure `acceptSA(·)`, on line 12, which implements the solution acceptance criteria from Simulated Annealing. In particular, the initial temperature, T_0, and the cooling factor, c, are calculated as being equal to $(\tau/100)(\hat{f}(S_{init})/(-\ln(0.5)))$ and $(T_0\,(-\ln(0.01)))^{(-1/K_{alns}^{max})}$, respectively, where $\hat{f}(S_{init})$ is the weighted sum $\sum_{i=3}^{9} w_i f_i(S_{init})$ of the number of violations, $f_i(S_{init})$, of every constraint C_i, for $i \in \{3, 4, \ldots, 9\}$, in the initial solution, S_{init}. The other parts of Algorithm 1 keep track of the best-found solution, S_{best}, and adjust the scores of the operators, which are used to recalculate the operator weights (and thus their selection probabilities) at the end of each K_{seg} iterations defining an ALNS segment.

Algorithm 1: ALNS for the NHL Scheduling Problem

parameters: K_{seg}, $\sigma_{bes}, \sigma_{bet}, \sigma_{acc}$, ρ, K^{max}_{alns}, τ, I^-, J^+, $\omega_o \; \forall o \in I^- \cup J^+$

$S_{init} \leftarrow$ `constructInitialSolution()`
foreach $o \in I^- \cup J^+$ **do**
    $\pi_o \leftarrow 0; \quad \theta_o \leftarrow 0$
$T_0 \leftarrow (\tau/100)(\hat{f}(S_{init})/(-\ln(0.5)))$;
$c \leftarrow (T_0\,(-\ln(0.01)))^{(-1/K^{max}_{alns})}$
$k \leftarrow 0; \quad k_{seg} \leftarrow 0$
$S_{curr} \leftarrow S_{init}; \quad S_{best} \leftarrow S_{init}$
while $k < K^{max}_{alns}$ ***and*** $f(S_{best}) > 0$ **do**
    $i \leftarrow$ `selectOperator(`$\omega_o \; \forall o \in I^-$`)`
    $j \leftarrow$ `selectOperator(`$\omega_o \; \forall o \in J^+$`)`
    $S_{temp} \leftarrow$ `applyOperators(`i, j, S_{curr}`)`
    if `acceptSA(`$f(S_{temp}) - f(S_{curr}), T_k$`)` **then**
        $S_{curr} \leftarrow S_{temp}$
        $\sigma' \leftarrow \sigma_{acc}$
    else
        $\sigma' \leftarrow 0$
    if $f(S_{temp}) < f(S_{best})$ **then**
        $S_{best} \leftarrow S_{temp}$
        $\sigma' \leftarrow \sigma_{bes}$
    else
        if $f(S_{temp}) < f(S_{curr})$ **then**
            $\sigma' \leftarrow \sigma_{bet}$
    $\pi_i \leftarrow \pi_i + \sigma'; \quad \pi_j \leftarrow \pi_j + \sigma'$
    $\theta_i \leftarrow \theta_i + 1; \quad \theta_j \leftarrow \theta_j + 1$
    if $k_{seg} < K_{seg}$ **then**
        $k_{seg} \leftarrow k_{seg} + 1$
    else
        $k_{seg} \leftarrow 0$
        foreach $o \in I^- \cup J^+ : \theta_o > 0$ **do**
            $\omega_o \leftarrow (1-\rho)\omega_o + \rho(\pi_o/\theta_o)$
            $\pi_o \leftarrow 0; \quad \theta_o \leftarrow 0$
    $T_{k+1} \leftarrow c\,T_k$
    $k \leftarrow k + 1$
return S_{best}

APPENDIX B

FLEURENT'S HEURISTICS

B.1 Fleurent's heuristics

In the heuristics by Fleurent (1987), a "feasibly possible" scheduling is a game-day assignment that satisfies all the constraints in the set $\mathsf{C}_F = \{\mathsf{C}_0, \mathsf{C}_1, \mathsf{C}_2, \mathsf{C}_4, \mathsf{C}_5, \mathsf{C}_6\}$, except perhaps C_5 on the minimum number of games per week.

B.1.1 Forced-trip heuristic

The forced-trip heuristic generates as many forced trips as feasibly possible. Every forced trip is generated by repeating the following description. At first, the heuristic scans all teams with at least two distant (more than 900 miles) away games to be scheduled, and chooses the longest period of consecutive days with neither arena-available nor already-assigned dates for a team. Let the chosen period be denoted by $[d_l, d_u]$ and the corresponding team by a. To anticipate, a will be the team to play away on the undertaken trip generation. In order to actually start building up the new trip, the heuristic selects two days, d_1 and d_2, both from within $[d_l, d_u]$, and two distant opponents, h_1 and h_2, that allows the team a to play away in the course of a single trip against h_1 and h_2 on days d_1 and d_2, respectively, without violating any constraint in C_F (except perhaps C_5 on the minimum number of games per week). If more than one choice is identified, the selection of those two days and two opponents is one that maximizes the saving measure value $s_a(h_1, h_2) = dist(h_1, a) + dist(a, h_2) - dist(h_1, h_2)$. It then carries out the corresponding game-day assignments for the visiting team, a, to create a trip $r(a)$, which is then formed by only two away games.

Next, the heuristic attempts to extend the trip $r(a)$ by trying to schedule the team a to directly visit other distant rivals both before having played against h_1 on day d_1 and after having played against h_2 on day d_2. In fact, it first tries to extend the end of the trip $r(a)$, whenever d_2 (the current last game day of $r(a)$) is far enough from $d_u + 1$ (the next arena-available date of team a) to possibly allow at least one more feasible away game to be played before a returns home, with no violation of the constraints in C_F. If this is the case and suitable rival exists, a team h_3 is selected as to maximize the saving value $s_a(h_2, h_3)$, and a is scheduled to visit h_3 on a day d_3, which now becomes the last game day of the trip $r(a)$. Such end-of-trip augmentation process is repeated until there exists no more distant rivals with arena dates yielding an away game scheduling for a at the end of the same trip, without

violating the constraints in C_F.

The process on extending the begin of the trip is essentially the same as that for the end. Indeed, the heuristic attempts to schedule new feasible away games for a by trying to select distant rivals which maximizes the saving measure defined above, but the course of the assignments now goes backward from d_1 (the current first game day of $r(a)$) to day d_l (the first non arena date for a on the selected period for the current trip generation).

B.1.2 Forced-home heuristic

The forced-home heuristic selects all the long-lasting trips in decreasing order of their duration, and tries to schedule free games by repeating the following steps for each trip $r(t)$ where a team t starts playing a game on day d_1^r and finishes playing some other game on day d_2^r, such that the condition $d_2^r - d_1^r \geq 7$ holds.

Step 1. *Schedule a home game before the current long-lasting trip.* At first, regarding only the arena dates of the team t that are before the day d_1^r, identify the day d_1 that is the closest day to d_1^r, and then, among the home games for t that can be scheduled on d_1 with no violation of constraints in C_F, schedule on d_1 the free game for which, if possible, the visiting rival of t does not have d_1 as one of its arena dates. In case of tie, prefer a rival located the closest to the arena of team t.

Step 2. *Schedule a home game after the current long-lasting trip.* At first, regarding only the arena dates of the team t that are after the day d_2^r, identify the day d_2 that is the closest day to d_2^r, and then, among the home games for t that can be scheduled on d_2 with no violation of constraints in C_F, schedule on d_2 the free game for which, if possible, the visiting rival of t does not have d_2 as one of its arena dates. In case of tie, prefer a rival located the closest to the arena of team t.

B.1.3 Free-trip heuristic

The free-trip heuristic tries to schedule all remaining games that individually opposes two distant teams by repeating the following steps on generating a single trip with no more than three games, where any condition of feasibility refers to the constraints in C_F only:

Step 1. *Select a visiting team.* Choose a team a with the highest number of distant (more than 900 miles) away games remaining to be scheduled, and let $\tilde{T}_a$ be the set of the respective (distant) rivals.

Step 2. *Determine a sequence of away games.* Among the teams in $\tilde{T}_a$, select two rivals, h_1 and h_2, to be visited by a in a row, such that the saving measure value $s_a(h_1, h_2) = dist(h_1, a) + dist(a, h_2) - dist(h_1, h_2)$ is the maximum over all pairs of teams from $\tilde{T}_a$.

Identify, also, a third opponent $h_3 \in \tilde{T}_a \setminus \{h_1, h_2\}$ that provides the new largest saving value $s_a(h_3, h')$, where h' is any team, other than h_3, from $\tilde{T}_a$ (perhaps even h_1 or h_2); and let now r be the sequence of teams given by the following conditional definition: $r = (h_1, h_2, h_3)$ if $h' = h_2$; or $r = (h_3, h_1, h_2)$ if $h' = h_1$; or $r = (h_1, h_2)$ otherwise.

Step 3. *Assign dates to the games.* Recursively try to schedule the team a to play away against its rivals in the sequence r, as follows. Initially, assign any feasible day to the first corresponding game so as to define a single-game trip. Then, try to extend the trip by sequentially scheduling a to visit the other teams in the order they appear in r. Clearly, from the definition of a trip, any pair of those consecutive games must take place within a period of three days. At each attempt to schedule a game, if none of the potential days allows an assignment without violating the constraints in C_F then, before to proceed, backtrack by canceling the last successful scheduled game and trying to assign another feasible day to it. If the whole recursive approach does not find any feasible period for the current free trip then apply it to: either the sequence r in inverse order if it has not been tried before; or a modified sequence in which the team geographically closest to a is discarded. On the contrary, determine all possible periods of assignment for the current trip and schedule its respective games to the period having the least number of arena dates for the travelling team, a.

B.1.4 Weekend-game heuristic

For each pair of a team t and a weekend day d, let the "cost" of scheduling t to play away on day d be defined as $c(t, d) = 3\delta_{td} + 2\delta_{td'} + 2\delta_{td''}$, where d' and d'' are the other two days (aside from d) in the weekend of d, and $\delta_{t\hat{d}}$ is equal to 1 if the day $\hat{d}$ is an arena date of team t, and it is equal to 0 otherwise. The weekend-game heuristic initially carries out the steps below for the set D of the regular season days that fall on a Saturday.

Step 1. *Create a set of promising games for every target day.* For each day $d \in D$, generate the set $\tilde{G}_d$ of every free game $\tilde{a}@\tilde{h}$ where a team $\tilde{a}$ must visit a non-distant (not farther than 900 miles) rival $\tilde{h}$ and such that all the following conditions are satisfied: *(1)* d yields an assignment with no violation of the constraints in C_F; *(2)* d is not within any long-distance trip period for neither $\tilde{a}$ nor $\tilde{h}$; and *(3)* d is either unavailable for $\tilde{a}$ or it is the only day available for $\tilde{a}$ among the three days of the corresponding weekend.

Step 2. *Sequentially schedule games to target days for each level of preference regarding the visiting team availability.* Whenever possible, select (at random if more than one) a target day $d \in D$ and a free game $\tilde{a}@\tilde{h}$ in $\tilde{G}_d$ such that $c(\tilde{a}, d)$ is the minimum over all dates in D and their respective promising games; remove $\tilde{a}@\tilde{h}$ from $\tilde{G}_d$; and assign d to

$\tilde{a}@\tilde{h}$ if it does not violate any constraint in C_F.

The heuristic repeats then these same steps two more times over different target days of the regular season period: first for D as the Fridays, and finally, for D as the Sundays.

B.1.5 Weekday-game heuristic

The weekday-game heuristic carries out the following steps, where any condition of feasibility refers to the constraints in C_F only:

Step 1. *Generate a set of feasible days for the free games.* Initially, create the set $\tilde{G}$ of all free games between non-distant (not farther than 900 miles) rivals. Then, for each $\tilde{a}@\tilde{h} \in \tilde{G}$, generate the set $D(\tilde{a}@\tilde{h})$ of feasible weekdays (from Monday through Thursday) which are not within any long-distance trip period for neither $\tilde{a}$ nor $\tilde{h}$.

Step 2. *Define the costs of the potential assignments.* Compute, for each pair of a free game $\tilde{a}@\tilde{h} \in \tilde{G}$ and a feasible day $d \in D(\tilde{a}@\tilde{h})$, a *game-day assignment cost* $c([\tilde{a}@\tilde{h}\colon d])$ as $n_H(d) + M$ if the arena of $\tilde{a}$ is available on day d, and as $n_H(d)$ otherwise, where $n_H(d)$ is the amount of teams that have d as one of their arena dates, and M is any big constant (the amount of teams in the League, for example).

Step 3. *Evaluate the regrets for the free games.* Compute, for each game $\tilde{a}@\tilde{h} \in \tilde{G}$, the regret $p(\tilde{a}@\tilde{h})$ as the difference between the costs $c([\tilde{a}@\tilde{h}\colon d''])$ and $c([\tilde{a}@\tilde{h}\colon d'])$, where d' and d'' are two days in $D(\tilde{a}@\tilde{b})$ corresponding to the first and to the second less costly assignments for $\tilde{a}@\tilde{h}$, respectively. If only one day happens to be feasible for a game then make the corresponding regret equal to the assignment cost.

Step 4. *Sequentially schedule games to weekdays.* Whenever possible, select (at random if more than one) a free game $\tilde{a}@\tilde{h} \in \tilde{G}$ such that the regret $p(\tilde{a}@\tilde{h})$ is the maximum over all free games in $\tilde{G}$; and assign the day d' to $\tilde{a}@\tilde{h}$, where d', which is involved in the computing of $p(\tilde{a}@\tilde{h})$, yields the first less costly assignment for $\tilde{a}@\tilde{h}$. After every assignment of a day d' to a game $\tilde{a}@\tilde{h}$, update the costs and the regrets for each game in $\tilde{G}$ that involves either $\tilde{a}$ or $\tilde{h}$ and for which the day d' is also implicated in one of its respective (two) less costly assignments.

B.1.6 Exchange heuristics

Both the single and the double exchange heuristics are presented as a recursive procedure on which every call to itself *returns* an output status indicating whether or not the respective game scheduling has been achieved. Given a partial schedule, the following two definitions regarding the potential assignment of a day d to a free game $\tilde{a}@\tilde{h}$ must be considered beforehand. We say that any scheduled game on day d is a C_0-*conflicting game for*

the assignment $[\tilde{a}@\tilde{h}: d]$ if, and only if, the scheduled game involves at least one of the teams $\tilde{a}$ and $\tilde{h}$. Moreover, given a non-negative integer $q \leq 2$, a day d is a *q-interchange feasible day for $\tilde{a}@\tilde{h}$* if *(1)* there is no conflicting game for $[\tilde{a}@\tilde{h}: d]$, but if so, *(2)* there are at most q conflicting games for $[\tilde{a}@\tilde{h}: d]$ and they are scheduled to that same day, d. Naturally, when a day d is a q-interchange feasible day for $\tilde{a}@\tilde{h}$, no constraint of the problem is violated if d is assigned to the game $\tilde{a}@\tilde{h}$ by first canceling all respective C_0-conflicting games that d may contain. The single and the double exchange heuristics, therefore, schedule games respectively to 1-interchange and to 2-interchange feasible days. Indeed, whether $q = 1$ or $q = 2$, the corresponding exchange heuristic can be seen as a recursive procedure that for each free game $\tilde{a}@\tilde{h}$ carries out the following steps, where l_{max} is a given maximum recursive level and any condition of feasibility refers to the constraints in C_F only:

Step 1. *Try to find a promising day d for the game $\tilde{a}@\tilde{h}$.* If all q-interchange feasible days for the game $\tilde{a}@\tilde{h}$ have already been considered (on possible previous "calls" to the current procedure) then *return "fail"*; otherwise, set d as any day selected at random among all the q-interchange feasible days for the game $\tilde{a}@\tilde{h}$ that minimize the respective number of conflicting games and that have not yet been selected for $\tilde{a}@\tilde{h}$. Go to Step 2.

Step 2. *Assign the day d to the game $\tilde{a}@\tilde{h}$ if it is feasible.* Schedule the game $\tilde{a}@\tilde{h}$ on the day d if there is no conflicting game for $[\tilde{a}@\tilde{h}: d]$ and *return "success"*; otherwise, go to Step 3.

Step 3. *Schedule $\tilde{a}@\tilde{h}$ on d and cancel any respective C_0-conflicting game if a new recursive level is allowed.* If $l = l_{max}$ then *return "fail"*; otherwise, let $G'([\tilde{a}@\tilde{h}: d])$ be the set of C_0-conflicting games for $[\tilde{a}@\tilde{h}: d]$, cancel every C_0-conflicting game $a@h \in G'([\tilde{a}@\tilde{h}: d])$, assign d to $\tilde{a}@\tilde{h}$, and then go to Step 4.

Step 4. *Recursively call the current procedure for the cancelled C_0-conflicting games for $[\tilde{a}@\tilde{h}: d]$.* For each cancelled game $a@h \in G'([\tilde{a}@\tilde{h}: d])$, call the current procedure (go to Step 1) with $\tilde{a}@\tilde{h}$ and l being replaced by $a@h$ and $l + 1$, respectively. After all those calls have finished (by a "*return*"), go to Step 5.

Step 5. *Undo all changes that might have been done by the Step 4 if it has failed to reschedule any of the cancelled C_0-conflicting games for $[\tilde{a}@\tilde{h}: d]$.* If every recursive call from the Step 4 has returned *success* as final status then *return "success"*; otherwise, remove the game $\tilde{a}@\tilde{h}$ from the day d and undo all changes (new assignments and new cancellations) that might have been done on the partial schedule by the Step 4 when trying to reschedule some C_0-conflicting game in $G'([\tilde{a}@\tilde{h}: d])$, and then go to Step 1.

APPENDIX C

DESTROY OPERATORS

C.1 Destroy operators

Let f_{-} be a function that evaluates the extended incremental cost of removing a game-day assignment $[a@h\colon d]$ from a schedule S as

$$f_{-}(a,h,d) = 5\, f_{-}^{\Delta}(a,h,d) + \bar{A}_{a,h,d},$$

where $f_{-}^{\Delta}(a,h,d)$ is the corresponding *removal incremental cost*, namely, $f(S \setminus \{[a@h\colon d]\}) - f(S)$; and $\bar{A}_{a,h,d}$, which we refer to as the *arena-utility value* for having the day d assigned to the game $a@h$, is a constant given by

$$\bar{A}_{a,h,d} = \begin{cases} 1 & \text{if } d \text{ is provided by } a \text{ but not by } h; \\ 2 & \text{if } d \text{ is provided neither by } a \text{ nor by } h; \\ 3 & \text{if } d \text{ is provided both by } a \text{ and by } h; \\ 4 & \text{if } d \text{ is provided by } h \text{ but not by } a. \end{cases}$$

Regarding this evaluation, the lower is the value $f_{-}(a,h,d)$, the "worse" is the assignment $[a@h\colon d]$, so the better is its removal from S. In addition, let $f_{+}^{\Delta}(a,h,d)$ be the *insertion incremental cost* for assigning a day d to a game $a@h$, namely, $f(S \cup \{[a@h\colon d]\}) - f(S)$.

Given a schedule S with n game-day assignments, and the integers $\hat{n}$ and $\tilde{n}$, such that $0 < \tilde{n} < \hat{n} \leq n$, the destroy operators in our ALNS algorithm perform the instructions that we informally state below.

C.1.1 Random-based removal

Random-based removal ($\mathsf{01}^{-}$) Initially, form a set $\hat{S}$ of target game-day assignments by choosing at random $\hat{n}$ assignments from the schedule S. Then, remove from S the $\tilde{n}$ target assignments in $\hat{S}$ with the lowest extended incremental cost determined by the values of f_{-}.

C.1.2 Structure-based removals

Divisional removal ($\mathsf{02}^-$) Initially, select at random an NHL division, T'_D. Then, form a set $\hat{S}$ of target game-day assignments by choosing at random $\hat{n}$ assignments from the current schedule, S, such that each chosen assignment involves at least one team from the division T'_D. Finally, remove from S the $\tilde{n}$ target assignments in $\hat{S}$ with the lowest extended incremental cost determined by the values of f_-.

Intradivisional removal ($\mathsf{03}^-$) Initially, select at random an NHL division, T'_D. Then, form a set $\hat{S}$ of target game-day assignments by choosing at random $\hat{n}$ assignments from the current schedule, S, such that both teams on each chosen assignment are from the division T'_D. Finally, remove from S the $\tilde{n}$ "worst" target assignments, which are the assignments in $\hat{S}$ with the lowest values of f_-.

Interdivisional removal ($\mathsf{04}^-$) Initially, select at random two NHL divisions, T'_D and T''_D. Then, form a set $\hat{S}$ of target game-day assignments by choosing at random $\hat{n}$ assignments from the current schedule, S, such that each chosen assignment involves a team from the division T'_D and a team from the division T''_D. Finally, remove from S the $\tilde{n}$ target assignments in $\hat{S}$ with the lowest extended incremental cost determined by the values of f_-.

Conferential removal ($\mathsf{05}^-$) Initially, select at random an NHL conference, T'_C. Then, form a set $\hat{S}$ of target game-day assignments by choosing at random $\hat{n}$ assignments from the current schedule, S, such that each chosen assignment involves at least one team from the conference T'_C. Finally, remove from S the $\tilde{n}$ target assignments in $\hat{S}$ with the lowest extended incremental cost determined by the values of f_-.

Intraconferential removal ($\mathsf{06}^-$) Initially, select at random an NHL conference, T'_C. Then, form a set $\hat{S}$ of target game-day assignments by choosing at random $\hat{n}$ assignments from the current schedule, S, such that both teams on each chosen assignment are from the conference T'_C. Finally, remove from S the $\tilde{n}$ target assignments in $\hat{S}$ with the lowest extended incremental cost determined by the values of f_-.

Interconferential removal ($\mathsf{07}^-$) Initially, form a set $\hat{S}$ of target game-day assignments by choosing at random $\hat{n}$ assignments from the current schedule, S, such that the two teams on each chosen assignment are not from the same conference. Finally, remove from S the $\tilde{n}$ target assignments in $\hat{S}$ with the lowest extended incremental cost determined by the values of f_-.

C.1.3 Neighboring-based removals

Day-neighboring removal ($\mathsf{08}^-$) Initially, select at random a day d' from the regular season. Then, given a positive integer q, form a set $\hat{S}$ of target game-day assignments by selecting all assignments involving the days from d' to $d' + q$ on the current schedule, S. Finally, remove from S the $\tilde{n}$ target assignments in $\hat{S}$ with the lowest extended incremental cost determined by the values of f_-.

Place-neighboring removal ($\mathsf{09}^-$) Initially, select at random a day d' from the regular season and a team t' from the League. Then, given a positive integer q, form a set $\hat{S}$ of target game-day assignments by selecting the $\hat{n}$ assignments on the current schedule, S, involving the geographically closest arenas to the arena of the team t', and restrict to the period from d' to $d' + q$. Finally, remove from S the $\tilde{n}$ target assignments in $\hat{S}$ with the lowest extended incremental cost determined by the values of f_-.

C.1.4 Critical removals

Arena-critical removal ($\mathsf{10}^-$) Repeat the following three steps until at least $\tilde{n}$ games have been removed from the schedule S: *(1)* firstly, select at random and then cancel an assignment $[\tilde{a}@t' : d']$ having the minimum arena-utility value over all the current assignments; *(2)* similarly, select and then cancel some assignment $[t'@\tilde{h} : d'']$ having the minimum arena-utility value among the current assignments that has t' as the visiting team on a day d'' that, regarding the game $\tilde{a}@t'$, does not violate the constraint C_6 on the minimum span for similar consecutive visits; *(3)* eventually, select and remove up to two C_i-conflicting games (if any) for the possible assignment of d'' to $\tilde{a}@t'$, regarding every constraint C_i, with $i = 0, 1, 2, 3, 4$, in this order.

Sequential greedy removal ($\mathsf{11}^-$) Sequentially remove a total of $\tilde{n}$ games from the schedule S by cancelling, at each time, the game-day assignment with the currently lowest extended incremental cost determined by the value of f_-.

Cost-critical removal ($\mathsf{12}^-$) Repeat the following three steps until at least $\tilde{n}$ games have been removed from the schedule S: *(1)* firstly, select and then cancel some assignment $[\tilde{a}@\tilde{h} : \tilde{d}]$ with the lowest extended incremental cost determined by the value of f_- over all the current assignments; *(2)* secondly, among the days provided by the team $\tilde{h}$, select a C_6-feasible day d' for $\tilde{a}@\tilde{h}$ having the minimum number of conflicting games on S for the possible assignment of d' to $\tilde{a}@\tilde{h}$ and, in case of tie, leading to the minimum insertion incremental cost, which is given by the corresponding value of f_+^{Δ}; *(3)* eventually, select and remove up to two C_i-conflicting games (if any) for the possible assignment of d' to $\tilde{a}@\tilde{h}$, regarding every constraint C_i, with $i = 0, 1, 2, 3, 4$, in this order.

C.1.5 Trip-based removals

Short-trip removal ($\mathsf{13}^-$) Repeat the following three steps until at least $\tilde{n}$ games have been removed from the schedule S: *(1)* firstly, select at random a trip $\tilde{r}$ having the minimum number of games and remove all them from the schedule; *(2)* secondly, for each game $\tilde{a}@\tilde{h}$ removed from the trip $\tilde{r}$, select, among the days provided by the team $\tilde{h}$, a C_6-feasible day d' for $\tilde{a}@\tilde{h}$ having the minimum number of conflicting games on S for the possible assignment of d' to $\tilde{a}@\tilde{h}$ and, in case of tie, yielding the minimum incremental cost on the objective function, which is given by the corresponding value of f_+^{Δ}; *(3)* eventually, select and remove up to two C_i-conflicting games (if any) for the assignment of d' to $\tilde{a}@\tilde{h}$, regarding every constraint C_i, with $i = 0, 1, 2, 3, 4$, in this order.

Close-trips removal ($\mathsf{14}^-$) Initially, select from S two consecutive trips, $\tilde{r}_1$ and $\tilde{r}_2$, for the same visiting team, that minimize the number of days between the end of trip $\tilde{r}_1$ and the begin of $\tilde{r}_2$; Then, form a set of target game-day assignments by choosing the $\hat{n}$ current assignments with the minimum arena-utility value on S, among the games (for any pair of teams) scheduled to days between the begin of trip $\tilde{r}_1$ and the end of $\tilde{r}_2$; Finally, remove from S the $\tilde{n}$ target assignments in $\hat{S}$ with the lowest extended incremental cost determined by the values of f_-.

Scattered-trip removal ($\mathsf{15}^-$) Initially, given a positive integer q, select from S a trip $\tilde{r}$ with at least q days long and that maximizes the ratio of its number of days to its number of games; Then, form a set of target game-day assignments by choosing the $\hat{n}$ assignments with the minimum arena-utility value on S, among the games (for any pair of teams) scheduled to days between the begin and the end of the trip $\tilde{r}$; Finally, remove from S the $\tilde{n}$ target assignments in $\hat{S}$ with the lowest extended incremental cost determined by the values of f_-.

APPENDIX D

REPAIR OPERATORS

D.1 Repair operators

Let f_+ be a function that evaluates the extended incremental cost of inserting a game-day assignment $[a@h\colon d]$ in a schedule S as

$$f_+(a,h,d) = 5\, f_+^{\Delta}(a,h,d) - \bar{A}_{a,h,d},$$

where $f_+^{\Delta}(a,h,d)$ is the corresponding *insertion incremental cost*, namely, $f(S \cup \{[a@h\colon d]\}) - f(S)$; and $\bar{A}_{a,h,d}$ is the *arena-utility value* for having the day d assigned to the game $a@h$, as defined in Appendix C.1. Regarding this evaluation, the lower is the value $f_+(a,h,d)$, the "better" is the insertion of the assignment $[a@h\colon d]$ in the schedule S. In addition, let D_t be the set of arena dates provided by a team t, and regarding a schedule S, let a day d be called a *free day* for a game $a@h$ if neither the team a nor the team h is currently scheduled to play on day d of the schedule S.

Given a partial schedule and a set of free games, the repair operators in our ALNS algorithm perform the instructions that we informally state below.

D.1.1 Greedy-based insertions

Single-evaluation greedy insertion ($\mathsf{01}^+$) Initially, evaluate by $f_+$the extended incremental cost of each pair of free game $\tilde{a}@\tilde{h}$ and arena-feasible day $d \in D_{\tilde{h}}$ that is a free day for $\tilde{a}@\tilde{h}$ in the current schedule. Then, select in a random order all the free games and, for each selected game $\tilde{a}@\tilde{h}$, proceed as follows: *(1)* verify whether there are days from $D_{\tilde{h}}$ that are still free for $\tilde{a}@\tilde{h}$ in the current schedule, and if so, *(2)* select among them a day d' for which the respective cost evaluated at the beginning of the heuristic is the lowest, and *(3)* update the current schedule by assigning d' to $\tilde{a}@\tilde{h}$. Finally, if not all the free games have been scheduled then repeat this approach by considering all free days (instead of only the arena-feasible days, $D_{\tilde{h}}$) in the current schedule.

Updated-evaluation greedy insertion ($\mathsf{02}^+$) Select all the free games, one at a time, by first choosing a team $\tilde{h}$ with the maximum number of free home games and then choosing a team $\tilde{a}$ with the maximum number of free away games among which at least one is against $\tilde{h}$. Each time a game $\tilde{a}@\tilde{h}$ is selected, proceed as follows: *(1)* verify whether there

are days from $D_{\tilde{h}}$ that are still free for $\tilde{a}@\tilde{h}$ in the current schedule, and if so, *(2)* select among them a day d' leading to the lowest extended incremental cost determined by the value of $f_+$with respect to the current schedule, and *(3)* update the schedule by assigning d' to $\tilde{a}@\tilde{h}$. If not all the free games have been scheduled then repeat this approach by considering all free days (instead of only the arena-feasible days, $D_{\tilde{h}}$) in the current schedule.

D.1.2 Regret-based insertion

Let $f_+^A(a, h, d)$ be a function that evaluates the cost of scheduling a free game $a@h$ to a corresponding arena-feasible date d that is free on a partial schedule S as

$$f_+^A(a, h, d) = M_1\, A_{a,d} + M_2\, f_+^{\Delta}(a, h, d) + |\tilde{G}(d)|,$$

where M_1 and M_2 are large constants, such that $M_1 \gg M_2$; $A_{a,d}$ is equal to 1 if the day d is also provided by the away team a, and equal to 0 otherwise; $f_+^{\Delta}(a, h, d)$ is the incremental cost of assigning d to $a@h$, namely, $f(S \cup \{[a@h\colon d]\}) - f(S)$; and $|\tilde{G}(d)|$, which can be seen as a "competitiveness" factor for d, is the current number of free games that are arena-feasible (it would satisfy C_1) for the day d.

Max-regret insertion ($\mathsf{03}^+$) Whenever there is still any free game, proceed as follows: *(1)* evaluate the *regret value* of each free game as the difference between its two lowest costs of f_+^A, which are provided by two corresponding arena-feasible dates that turns out to be free days in the current schedule; *(2)* select a free game $\tilde{a}@\tilde{h}$ for which the regret value is the maximum over all free games; and *(3)* update the current schedule by assigning to $\tilde{a}@\tilde{h}$ the respective lowest-cost day (with regard to the function f_+^A).

D.1.3 exchange-based insertion

Let $\tilde{a}@\tilde{h}$ be a free game and let $D'(\tilde{a}@\tilde{h})$ be its set of *1-exchange feasible days*, which are either feasible days for $\tilde{a}@\tilde{h}$ or days that would become feasible by first removing a single (conflicting) game currently scheduled on any day. If feasible days are available, the current free game is scheduled to a day $\hat{d} \in D'(\tilde{a}@\tilde{h})$ that maximizes the corresponding arena-utility value, $\bar{A}_{\tilde{a},\tilde{h},\hat{d}}$; otherwise, only days that would result in a single conflicting game are considered. In this case, the current free game is assigned to a day $\hat{d} \in D'(\tilde{a}@\tilde{h})$ that maximizes, over all 1-exchange feasible days, the respective arena-utility difference, $\bar{A}_{\tilde{a},\tilde{h},\hat{d}} - \bar{A}_{a',h',d'}$, where the game $a'@h'$, currently scheduled on a day d' (which is not necessarily equal to $\hat{d}$), is the conflicting game for the assignment of $\hat{d}$ to $\tilde{a}@\tilde{h}$.

Single-exchange insertion ($\mathsf{04}^{+}$) Select all the free games, one at a time, in decreasing order of their quantity for the same *away-home* pair of teams. Try to insert each selected game $\tilde{a}@\tilde{h}$ in the current schedule, S, by calling the *single-exchange procedure* while considering the 1-exchange feasible days with regard to all the constraints of the problem and selecting the most promising day as the one that *(1)* maximizes the arena-utility value over feasible days (if any) or *(2)* maximizes the difference between the arena-utility value for a 1-exchange feasible day and the arena-utility value for the corresponding conflicting game in S (otherwise). Eventually, apply the strategy from *Updated-evaluation greedy insertion* ($\mathsf{02}^{+}$) whenever no 1-exchange feasible day exists for a free game.

D.1.4 Fleurent's approach-based insertion

Fleurent's approach-based insertion ($\mathsf{05}^{+}$) Initially, apply the *forced-trip heuristic* but, instead of building one entire trip at a time, proceed as follows. At first, create trips made up of exactly two long-distance visits, such that each visit is scheduled during a period with no other game between two consecutive arena-available dates for the corresponding away team. After having created as many of such "partial" trips as possible, try to extend the current long-distance trips by only scheduling free games that define long-distance visits. Subsequently, apply the *free-trip heuristic* with no modifications. Then, apply the *weekend-game heuristic* to all the free games (not only those concerning long-distance visits) by trying to schedule them on any free weekend day that provides a feasible assignment, even if it is a day within a period of a long-distance trip for the respective teams. Finally, apply the *weekday-game heuristic* to all the free games (not only those concerning long-distance visits) by trying to schedule them to any free weekday that provides a feasible assignment, even if it is a day within a period of a long-distance trip for the respective teams. In addition, replace the $n_H(d)$ from the *weekday-game heuristic*, which is the amount of teams that are available on a day d, by the number of free games that are actually feasible candidates for d at the moment of each evaluation. Eventually, apply the strategy from *Max-regret insertion* ($\mathsf{03}^{+}$) if there still is any free game.