

Population-Based Simulated Annealing for Traveling Tournaments*

Pascal Van Hentenryck and Yannis Vergados

Brown University, Box 1910, Providence, RI 02912

Abstract

This paper reconsiders the travelling tournament problem, a complex sport-scheduling application which has attracted significant interest recently. It proposes a population-based simulated annealing algorithm with both intensification and diversification. The algorithm is organized as a series of simulated annealing waves, each wave being followed by a macro-intensification. The diversification is obtained through the concept of elite runs that opportunistically survive waves. A parallel implementation of the algorithm on a cluster of workstations exhibits remarkable results. It improves the best known solutions on all considered benchmarks, sometimes reduces the optimality gap by about 60%, and produces novel best solutions on instances that had been stable for several years.

Introduction

Sport scheduling has become a steady source of challenging applications for combinatorial optimization. These problems typically feature complex combinatorial structures (e.g., round-robin tournaments with side constraints), as well as objective functions measuring the overall quality of the schedule (e.g., travel distance).

The travelling tournament problem (TTP) is an abstraction of Major League Baseball (MLB) proposed by Easton, Nemhauser, and Trick 2001 to stimulate research in sport scheduling. The TTP consists of finding a double round-robin schedule satisfying constraints on the home/away patterns of the teams and minimizing the total travel distance. The TTP has been tackled by numerous approaches, including constraint and integer programming (and their hybridizations) (Easton, Nemhauser, & Trick 2001), Lagrangian relaxation (Benoist, Laburthe, & Rottembourg 2001), and meta-heuristics such as simulated annealing (Anagnostopoulos *et al.* 2003; 2006), tabu search (Di Gaspero & Schaerf 2006), GRASP (Ribeiro & Urrutia 2007), and hybrid algorithms (Lim, Rodrigues, & Zhang 2006) to name only a few. The best solutions so far have been obtained by meta-heuristics, often using variations of the neighborhood proposed in (Anagnostopoulos *et al.* 2003). This includes

new variations of the TTP based on circular and NFL distances. It is also interesting to observe that progress seems to have slowed down in recent years and some of the early instances have not been improved for several years.

This paper presents a population-based simulated annealing algorithm with both intensification and diversification components. The core of the algorithm is organized as a series of waves, each wave consisting of a collection of simulated annealing runs. At the end of each wave, an intensification takes place: a majority of the runs are restarted from the best found solution. Diversification is achieved through the concept of *elite runs*, a generalization of the concept of elite solutions. At the end of each wave, the simulated annealing runs that produced the k-best solutions so far continue their execution opportunistically from their current states. This core procedure is terminated when a number of successive waves fail to produce an improvement and is then restarted at a lower temperature.

This population-based simulated annealing was implemented on a cluster of workstations. It produces new best solutions on all TTP instances considered, including the larger NLB instances which had not been improved for several years, the circular instances, and the NFL instances for up to 26 teams. Although simulated annealing is not the most appropriate algorithm for the circular instances, the population-based algorithm has improved all best solutions for 12 teams or more on these instances. The improvements are often significant, reducing the optimality gap by almost 40% on many instances. The parallel implementation also obtained these results in relatively short times compared to the simulated annealing algorithm in (Van Hentenryck & Vergados 2006).

The broader contributions of the paper are twofold. First, it demonstrates the potential complementarity between the macro-intensification and macro-diversification typically found in tabu-search and population-based algorithms and the micro-intensification (temperature decrease) and micro-diversification (threshold acceptance of degrading move) featured in simulated annealing. Second, the paper indicates the potential benefit of population-based approaches for simulated annealing, contrasting with recent negative results in (Onbaşoğlu & Özdamar 2001).

*Partially supported by NSF award DMI-0600384 and ONR Award N000140610607.

Copyright © 2007, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Problem Description

A TTP input consists of n teams (n even) and an $n \times n$ symmetric matrix d , such that d_{ij} represents the distance between the homes of teams T_i and T_j . A solution is a schedule in which each team plays with each other twice, once in each team's home. Such a schedule is called a *double round-robin tournament*. A double round-robin tournament thus has $2n - 2$ rounds. For a given schedule S , the cost of a team is the total distance that it has to travel starting from its home, playing the scheduled games in S , and returning back home. The cost of a solution is defined as the sum of the cost of every team. The goal is to find a schedule with minimal travel distance satisfying the following two constraints:

1. **Atmost Constraints:** No more than three consecutive home or away games are allowed for any team.
2. **Norepeat Constraints:** A game of T_i at T_j 's home cannot be followed by a game of T_j at T_i 's home.

The Simulated Annealing Algorithm

This paper leverages the simulated annealing algorithm TTSA (Anagnostopoulos *et al.* 2006) which is used as a black-box. It is not necessary to understand its specificities which are described elsewhere. It suffices to say that TTSA explores a large neighborhood whose moves swap the complete/partial schedules of two rounds or two teams, or flip the home/away patterns of a game. The objective function f combines the total distance and the violations of the *atmost* and *norepeat* constraints. TTSA uses strategic oscillation to balance the time spent in the feasible and infeasible regions.

Population-Based Simulated Annealing

The core of the population-based simulated annealing receives a configuration S (e.g., a schedule) and a temperature T . It executes a series of waves, each of which consists of n executions of the underlying simulated annealing algorithm (in this case, TTSA). The first wave simply executes SA (S, T) N times (where N is the size of the population). Subsequent waves consider both opportunistic and intensified executions. The simulated annealing runs that produced the k -best solutions so far continue their executions: hopefully they will produce new improvements and they provide the macro-diversification of the algorithm. The $N - k$ remaining runs are restarted from the best solution found so far and the temperature T .

Figure 1 illustrates the core of the algorithm for a population of size $N = 20$ and $k = 4$. Figure 1(a) shows that all executions start from the same configuration and Figure 1(b) depicts the behavior during wave 1. The best solution obtained is $S_{8,1}^*$ (solid square in figure). Several other executions also produces solutions ($S_{2,1}^*$, $S_{5,1}^*$, $S_{10,1}^*$, $S_{13,1}^*$, $S_{18,1}^*$) that improve upon their starting points (circles in figure). The best 3 of them (solid circle), together with the best solution found so far, define the elite runs used for diversification (i.e., $S_{2,1}^*$, $S_{8,1}^*$, $S_{13,1}^*$, $S_{18,1}^*$). Figure 1(c) depicts the start of the second wave. It highlights that the elite runs continue their execution from their current state, while the remaining 16 executions restarts from the best solution $S_{8,1}^*$,

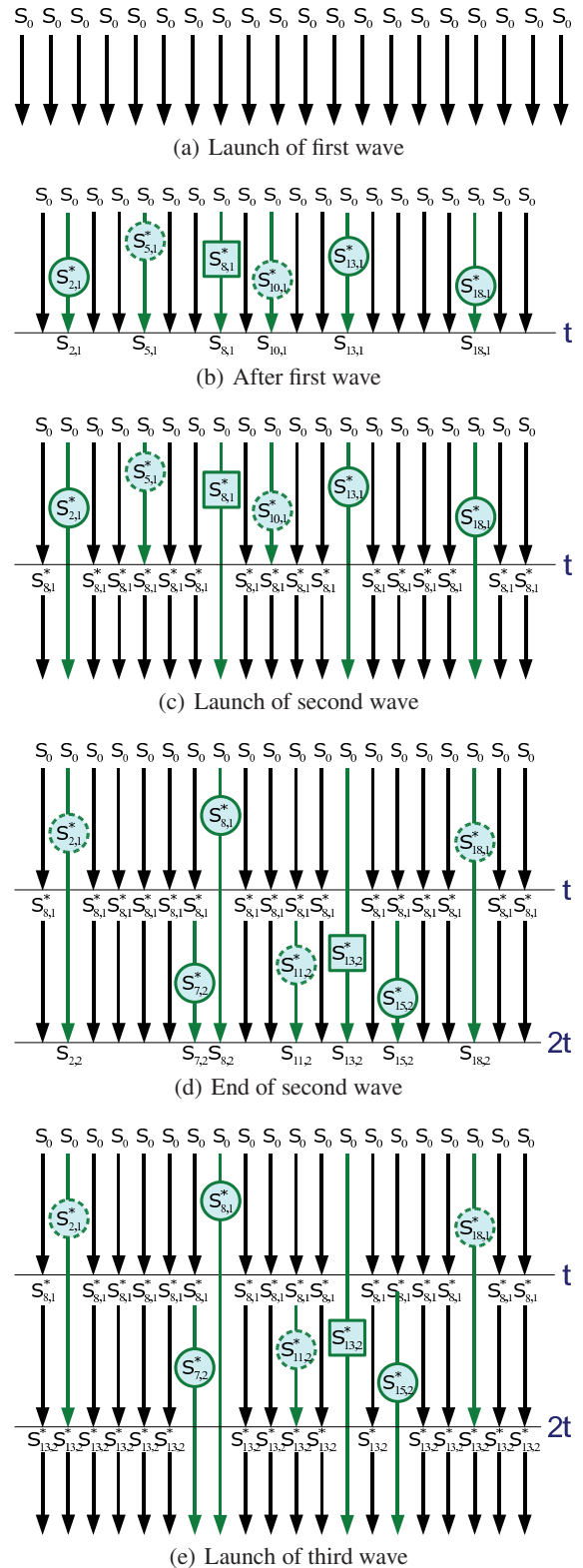


Figure 1: Illustrating PBSA with $k = 4$.

and the initial temperature. Figure 1(d) shows the executions of the first two waves. The second wave found a new best solution $S_{13,2}^*$ (produced by one of the elite runs), while several executions improve upon their starting points. The 4 best solutions are now $S_{7,2}^*$, $S_{8,1}^*$, $S_{13,2}^*$, and $S_{15,2}^*$, and the simulated annealing executions that produced them are now the set of elite runs. Figure 1(e) depicts the launch of the third wave. Observe that the two elite runs (those that produced $S_{8,1}^*$ and $S_{13,2}^*$) will now execute for the third successive wave, while two new ones have emerged. This core procedure terminates after a number of stable waves, i.e., successive waves that have not improved the best solution. It is embedded in an outermost loop that progressively decreases the temperature T .

The overall algorithm is depicted in Figures 2 and 3. Figure 2 describes the core procedure PBSA-P for a population \mathcal{P} of size $N = |\mathcal{P}|$. For each member p of the population, the algorithm maintains its current starting configuration S_p and temperature T_p , as well as the value f_p of the best solution p has generated. These variables are initialized in lines 2–6. The algorithm also maintains the best overall solution S^* and the number *stable* of successive waves without improvement to S^* . Lines 8–23 are concerned with the execution of a wave. For each $p \in \mathcal{P}$, PBSA-P applies the simulated annealing algorithm for t units of time on configuration S_p with starting temperature S_p . The simulated annealing execution returns the best solution S_p^* of this run and the final configuration S_p^+ and temperature T_p^+ (line 8). If the run improves its starting solution, i.e., $f(S_p^*) < f(S_p)$, PBSA-P updates variable f_p (line 11). If these runs have not improved the best solutions, the next wave continues each of the runs from their current state (lines 12–13). Otherwise, the runs that produced the k -best solutions (the elite runs) continue their executions, while the remaining $N - k$ runs (the set \mathcal{R} in line 19) are restarted from their current best solution S^* and the initial temperature T (line 20–22). Figure 3 shows that PBSA-P is embedded in a loop which progressively decreases the temperature (lines 3–6). The overall algorithm PBSA also starts from a solution produced by simulated annealing or, possibly, any other algorithm.

Experimental Results

Experimental Setting The algorithm was implemented in parallel to execute each run in a wave concurrently. The experiments were carried out on a cluster of 60 Intel-based, dual-core, dual-processor Dell Poweredge 1855 blade servers. Each server has 8GB of memory and a 300G local disk. Scheduling on the cluster is performed via the Sun Grid Engine, version 6. The tested instances are the non-mirrored NLB n , CIRC n and NFL n instances described in (Trick 2002).

The experiments use a population of size $N = 80$ and the number k of elite runs is in the range [10,30]. The time duration t of each wave is in the range of [60,150] seconds depending on the size of the instances. PBSA-P terminates after a maximum number of successive non-improving waves chosen in the range of [5,10]. PBSA is run for 10 phases with $\beta = 0.96$. We report two types of results starting from

```

1. function PBSA-P(S, T) {
2.   forall  $p \in \mathcal{P}$  do
3.      $S_p = S$ ;
4.      $f_p = f(S)$ ;
5.      $T_p = T$ ;
6.    $S^* = S$ ;
7.   stable = 0;
8.   while stable < maxStableWaves do
9.     forall  $p \in \mathcal{P}$  do
10.       $\langle S_p^*, S_p^+, T_p^+ \rangle = \text{SA}_t(S_p, T_p)$ ;
11.      if  $f(S_p^*) < f(S_p)$  then  $f_p = f(S_p^*)$ ;
12.       $S_p = S_p^+$ ;
13.       $T_p = T_p^+$ ;
14.       $b = \arg \min_{p \in \mathcal{P}} f_p$ ;
15.      if  $f(S^*) > f(S_b^*)$  then
16.         $S^* = S_b^*$ ;
17.        stable = 0;
18.         $f^k = k - \min_{p \in \mathcal{P}} f_p$ ;
19.         $\mathcal{R} = \{p \in \mathcal{P} : f_p > f^k\}$ ;
20.        forall  $p \in \mathcal{R}$  do
21.           $S_p = S^*$ ;
22.           $T_p = T$ ;
23.        else stable = stable + 1;
24.      return  $S^*$ ;
25. }
```

Figure 2: PBSA-P: A Phase of PBSA

```

1. function PBSA(S) {
2.    $T \leftarrow T_0$ ;
3.    $S \leftarrow \text{SA}_t(S, T)$ ;
4.   for phase = 1 to maxPhases do
5.      $S \leftarrow \text{PBSA-P}(S, T)$ ;
6.      $T \leftarrow T \cdot \beta$ ;
7.   end for
8.   return  $S$ ;
9. }
```

Figure 3: The Algorithm PBSA

either low or high-quality TTSA solutions. All results reported are averages over 10 runs.

PBSA from High-Quality Solutions The experimental results are summarized in Tables 1, 2, and 3, which report both on solution quality and execution times. With respect to solution quality, the tables describe the previous best solution (best) (not found by PBSA), the best lower bound (LB), the minimum (min) and average (mean) travel distances found by PBSA, and the improvement in the optimality gap (best - LB) in percentage (%G). The results on execution times report the times (in seconds) taken for the best run (Time(Best)), the average times (mean(T)), and the standard deviation (std(T)).

As far as solution quality is concerned, PBSA improves on all best-known solutions for the NLB and circular instances with 14 teams or more. These NLB instances had

n	Best	LB	min	mean	%G
14	189156	182797	188728	188728.0	6.7
16	267194	249477	262343	264516.4	27.3

n	Time(Best)	mean(T)	std(T)
14	360	264.0	139.94
16	600	468.0	220.94

Table 1: Quality and Times for NLB Distances.

n	Best	LB	min(D)	mean(D)	%G
12	408	384	408	414.8	0
14	654	590	632	645.2	34.3
16	928	846	916	917.8	14.6
18	1306	1188	1294	1307.0	10.1
20	1842	1600	1732	1754.4	45.4

n	Time(Best)	mean(T)	std(T)
12	1440	648.0	630.88
14	1080	402.0	287.81
16	180	342.0	193.58
18	4680	3380.0	1950.86
20	10270	8437.0	1917.18

Table 2: Quality and Times for Circular Distances.

not been improved for several years despite new algorithmic developments and approaches. It also improves the NFL instances for 16 to 26 teams (larger instances were not considered for lack of time). The improvement in the optimality gap is often substantial. For NLB-16, CIRC-20, and NFL-20, the improvements are respectively about 27%, 45%, and 48%.

As far as solution times are concerned, PBSA typically finds its best solutions in times significantly shorter than TTSA. On the NLB instances, PBSA found its new best solutions within 10 minutes, although these instances had not been improved for a long time. Typically, the new best solutions are found within an hour for problems with less than 20 teams and in less than two hours otherwise. These results are quite interesting as they exploit modern architectures to find the best solutions in competitive times, the elapsed times being significantly shorter than TTSA.

PBSA from Scratch Figures 4, 5, and 6 describe the performance of PBSA when the TTSA is only run shortly to produce a starting point. These results are particularly interesting. PBSA improves the best known solutions for the NLB instances for 12, 14, and 16 teams, for the circular instances 12 and 14, and for NFL instances 16, 18, 20, 22, and 26.

For the NLB, the improvement for 14 teams is the same as when PBSA starts from a high-quality solution, while the improvement for 12 and 16 teams is even better, producing new best solutions. The optimality gap is reduced by about

n	Best	LB	min(D)	mean(D)	%G
16	235930	223800	231483	232998.4	36.6
18	296638	272834	285089	286302.9	48.5
20	346324	316721	332041	332894.5	48.2
22	412812	378813	402534	404379.7	30.2
24	467135	-	463657	465568.7	-
26	551033	-	536792	538528.0	-

n	Time(Best)	mean(T)	std(T)
16	2220	1356.0	998.31
18	3120	2412.0	1811.52
20	6750	4419.0	1349.06
22	8100	4365.0	2484.79
24	5490	4113.0	2074.70
26	6480	3024.0	1927.42

Table 3: Quality and Times for NFL Distances.

n	Best	LB	min	mean	%G
12	111248	107494	110729	112064.0	13.8
14	189156	182797	188728	190704.6	6.7
16	267194	249477	261687	265482.1	31.0

n	Time(Best)	mean(T)	std(T)
12	2370	1501.5	816.73
14	3045	2491.5	1067.94
16	18150	12858.0	3190.31

Table 4: Quality and Times in Seconds for NLB Distances Starting from Scratch.

14%, 7%, and 31% respectively. For the circular instances, the improvement is better than when PBSA starts from a high-quality solution, for 12 teams, but not as good for more teams. In fact, starting from scratch improves upon the best known solution only for $n = 14$, and is not very competitive for larger n . Finally, for the NFL, it is interesting to note that, for $n = 18$, PBSA produces a better solution starting from scratch than from a high-quality solution.

TTSA versus PBSA Figure 4 depicts the behavior of TTSA over a long time period (three days) and compares it with PBSA. In this experiment, 80 independent TTSA processes run concurrently with no information exchange and the figure shows the evolution of the best found solution over time. The results show that TTSA achieves only marginal improvement after the first few hours. However, in about five hours, PBSA achieves a substantial improvement over the best solution found by TTSA in the three days.

The Effect of Macro-Diversification To assess the benefits of macro-diversification, we ran PBSA (for 10 iterations) on NLB-16 starting from scratch, varying the number of elite runs k . Table 7 depicts the minimum and the mean cost and the gap reduction. The results seem to indi-

n	Best	LB	min	mean	%G
12	408	384	404	418.2	16.6
14	654	590	640	654.8	21.8
16	928	846	958	971.8	-36.5
18	1306	1188	1350	1371.6	-37.2
20	1842	1600	1866	1886.2	-9.9

n	Time(Best)	mean(T)	std(T)
12	2200	1102.0	560.31
14	1720	1396.0	457.10
16	7260	4962.0	1743.11
18	6660	5994.0	4070.67
20	8440	5157.5	2562.67

Table 5: Quality and Times for Circular Distances Starting From Scratch.

n	Best	LB	min	mean	%G
16	235930	223800	233419	234847.9	20.7
18	296638	272834	282258	285947.6	60.4
20	346324	316721	333429	337280.3	43.5
22	412812	378813	406201	412511.8	19.4
24	467135	-	471536	476446.6	-
26	551033	-	545170	553175.5	-

n	Time(Best)	mean(T)	std(T)
16	14010	14325.0	1626.11
18	19320	17097.0	2164.13
20	19680	18771.0	2053.67
22	23730	17778.7	4686.12
24	22110	18645.0	2150.32
26	18600	24621.0	3448.87

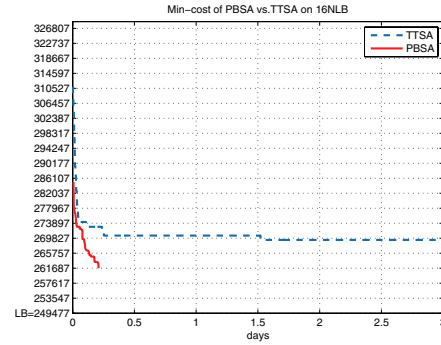
Table 6: Quality and Times for NFL Distances Starting From Scratch.

cate a nice complementarity between macro-intensification and macro-diversification.

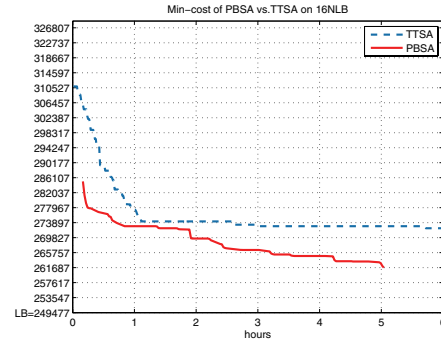
Related Work

Cooperative Parallel Search Population-based simulated annealing can be viewed as a cooperative parallel search. Onbařođlu et. al. 2001 provide an extensive survey of parallel simulated annealing algorithms and compare them experimentally on global optimization problems. They also classify those schemes into application-dependent and application-independent parallelization.

In the first category, the problem instance is divided among several processors, which communicate only to deal with dependencies. In the second category, Onbařođlu et. al. further distinguish between asynchronous parallelization with no processor communication, synchronous parallelization with different levels of communication, and highly-coupled synchronization in which neighborhood solutions are generated and evaluated in parallel. In the first two cases, processors work on separate Markov chains while, in the



(a) Min-Cost Graph on NLB-16



(b) Min-Cost Graph on NLB-16 (Zoomed)

Figure 4: Comparison of Min-Cost Evolution for TTSA and PBSA on NLB-16

k	Best	LB	min	mean	%G
0	267194	249477	266130	268538.6	6.0
10	267194	249477	264472	267261.0	15.3
20	267194	249477	263304	267563.1	21.9
30	267194	249477	261687	265482.1	31.0

Table 7: The Effect of Macro-Diversification (NLB-16).

third case, they cooperate on a single Markov chain.

PBSA can thus be viewed as an application-independent algorithm with synchronous parallelization and periodic exchange of solutions. The scheme proposed in (Janaki Ram, Sreenivas, & Ganapathy Subramaniam 1996) (which only exchanges partial solutions) and the SOEB-F algorithm (Onbařođlu & Özdamar 2001) are probably the closest to PBSA but they do not use diversification and elite runs. Observe also that SOEB-F typically fails to produce sufficiently good solutions (Onbařođlu & Özdamar 2001).

It is also useful to point out that the above classification is not limited to simulated annealing. A cooperative parallel scheme based on tabu search is presented in (Asahiro, Ishibashi, & Yamashita 2003) and is applied to the generalized assignment problem. In this context, we can point out PBSA could be lifted into a generic algorithm providing macro-intensification and macro-diversification for any meta-heuristic. Whether such a generic algorithm would be

useful in other contexts remains to be seen however.

Memetic Algorithms and Scatter Search PBSA can be seen as a degenerated form of scatter search (Laguna & Marti 2003) where solutions are not combined but only intensified. Moreover, the concept of elite solutions is replaced by the concept of elite runs which maintains the state of the local search procedures. PBSA can also be viewed as a degenerated form of memetic algorithms (Norman & Moscato 1991), where there is no mutation of solutions: existing solutions are either replaced by the best solution found so far or are “preserved”. Once again, PBSA does more than preserving the solution: it also maintains the state of the underlying local search through elite runs. It is obviously an interesting research direction to study how to enhance PBSA into an authentic scatter search and memetic algorithm. The diversification so-obtained may further improve the results.

Conclusions

This paper proposed a population-based simulated annealing algorithm for the TTP with both macro-intensification and macro-diversification. The algorithm is organized as a series of waves consisting of many simulated annealing runs. Each wave being followed by a macro-intensification restarting most of the runs from the best found solution and a macro-diversification which lets elite runs the chance to produce new best solutions.

The algorithm was implemented on a cluster of workstations and exhibits remarkable results. It improves the best known solutions on all considered benchmarks, may reduce the optimality gap by about 60%, and produces better solutions on instances that had been stable for several years. Moreover, these improvements were obtained by the parallel implementation in times significantly shorter than the simulated annealing algorithm TTSA, probably the best algorithm overall to produce high-quality solutions.

These results shed some light on the complementarity between the micro-intensification and micro-diversification inherent to simulated annealing and the macro-intensification and macro-diversification typically found in other metaheuristics or frameworks. Future research will be devoted to understand how to combine TTP solutions to produce scatter search and memetic algorithms for the TTP.

References

- Anagnostopoulos, A.; Michel, L.; Van Hentenryck, P.; and Vergados, Y. 2003. A Simulated Annealing Approach to the Traveling Tournament Problem. In *CP-AI-OR'03*.
- Anagnostopoulos, A.; Michel, L.; Van Hentenryck, P.; and Vergados, Y. 2006. A Simulated Annealing Approach to the Traveling Tournament Problem. *Journal of Scheduling* 9:177–193.
- Asahiro, Y.; Ishibashi, M.; and Yamashita, M. 2003. Independent and cooperative parallel search methods. *Independent and Cooperative Parallel Search Methods* 18(2):129–141.
- Benoist, T.; Laburthe, F.; and Rottembourg, B. 2001. Lagrange relaxation and constraint programming collaborative schemes for travelling tournament problems. In *CPAIOR 2001*, 15–26.

- Di Gaspero, L., and Schaerf, A. 2006. A composite-neighborhood tabu search approach to the traveling tournament problem. *Journal of Heuristics*. To appear.
- Easton, K.; Nemhauser, G.; and Trick, M. 2001. The Traveling Tournament Problem Description and Benchmarks. In *CP'01*, 580–589. Springer-Verlag.
- Janaki Ram, D.; Sreenivas, T. H.; and Ganapathy Subramaniam, K. 1996. Parallel simulated annealing algorithms. *J. Parallel Distrib. Comput.* 37(2):207–212.
- Laguna, M., and Marti, R. 2003. *Scatter Search*. Kluwer Academic Publishers.
- 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(3):1459–1478.
- Norman, M., and Moscato, P. 1991. A competitive and cooperative approach to complex combinatorial search. In *20th Informatics and Operations Research Meeting*.
- Onbaşoğlu, E., and Özdamar, L. 2001. Parallel simulated annealing algorithms in global optimization. *J. of Global Optimization* 19(1):27–50.
- Ribeiro, C., and Urrutia, S. 2007. Heuristics for the mirrored traveling tournament problem. *European Journal of Operational Research* 179(3):775–787.
- Trick, M. 2002–2006. Challenge traveling tournament problems. <http://mat.gsia.cmu.edu/TOURN/>.
- Van Hentenryck, P., and Vergados, Y. 2006. Traveling tournament scheduling: A systematic evaluation of simulated annealing. In *CPAIOR 2006*, 228–243. Springer.