A Tabu Search Algortihm for the Railway Scheduling Problem

Dario Pacciarelli[*] *Marco Pranzo*[*]

[*] Dipartimento di Informatica e Automazione, Università di Roma Tre
Via della Vasca Navale, 79, 00146 Roma, Italy
Email: pacciarelli@dia.uniroma3.it, mpranzo@dia.uniroma3.it

1 Introduction

Aim of this presentation is to introduce a tabu search algorithm for a general scheduling problem modeled by means of the alternative graph. The alternative graph is a mathematical model, which is based on a generalization of the disjunctive graph of Roy and Sussman [6]. It allows a detailed representation of many scheduling problems. It was successfully applied to several real world environments such as the production of stainless steel and traffic management of railway networks.

We next introduce the alternative graph formulation, and then formulate a railway scheduling problem by means of the alternative graph. Finally we describe our local search procedure.

2 The alternative graph

To model the train scheduling problem we use the alternative graph formulation introduced by Mascis and Pacciarelli [5], which is based on a generalization of the disjunctive graph. The alternative graph formulation can be viewed as a particular disjunctive program, i.e. a linear program with logical conditions involving operations "and" ($\wedge$, conjunction) and "or" ($\vee$, disjunction), as in Balas [2].

$$\min t_n$$
$$\begin{cases} t_j - t_i \geq f_{ij} & (i,j) \in F \\ (t_j - t_i \geq a_{ij}) \vee (t_k - t_h \geq a_{hk}) & ((i,j),(h,k)) \in A \end{cases}$$

The *alternative graph* is described by a triple $G = (N, F, A)$. There is a set of nodes $N = \{0, 1, \ldots, n\}$, a set of directed arcs F and a set of pairs of directed arcs A. Arcs in the set F are fixed. Arcs in the set A are *alternative.* If $((i,j),(h,k)) \in A$, we say that (i,j) and (h,k) are paired and that (i,j) is the alternative of (h,k). In our model the arc length can be either positive, null or negative. A selection S is a set of arcs obtained from A by choosing at most one arc from each pair. The selection is *complete* if exactly one arc from each pair is chosen. Given a selection S let $G(S)$ indicates the graph $(N, D \cup S)$. A selection S is consistent if the graph $G(S)$ has no positive length cycles. Given a consistent selection S, we call extension of S a complete consistent selection S' such that $S \subseteq S'$, if it exists. The *makespan* of a consistent selection S is defined as the length of a longest path from node 0 to node n in $G(S)$.

3 Railway scheduling

In its basic form a railway network is composed by track lines and signals. There are signals before every station, passing loop, junction, etc., as well as along the lines. A *block section* is a track segment between two signals. A signal may be in one of several conditions, say either red, yellow, or green. A red signal means that the subsequent block section is occupied by another train, a yellow signal means that the subsequent block section is empty, but the following one is occupied by another train, and a green signal means that the next two block sections are empty. A train is allowed to enter a block section depending both on its speed and on the signal color. Slow trains can enter a block section only if the signal is green or yellow, fast trains can enter a block section at high speed only if the signal is green. Hence, each section can host at most one train at a time.

Stopping or slowing a train causes a remarkable loss of time and energy, due to the long braking distances, followed by acceleration of large masses. More important, if a railway line slopes up over a certain gradient, then there are some freight trains that should not decrease their speed under a certain limit, otherwise they would not be able to reach the top, due to the lack of horsepower. Therefore, in a feasible schedule, there are some freight trains that must always find green or yellow signals, and however, in a good schedule, fast trains should always find green signals, and slow trains should always find green or yellow signals.

A block section takes a minimum time to be covered, which is known in advance for each train and which is used to develop the Master Schedule. All trains travel at their planned speed unless they are delayed. In such cases they travel at their maximum speed, whenever possible, in order to recover the delay.

We use the alternative graph formulation to model the train scheduling problem and we show that each feasible schedule for the trains can be associated with a complete consistent selection on a suitable alternative graph. The problem of minimizing the maximum delay for all trains at all stations can be therefore modeled as the problem of finding a minimum makespan complete consistent selection.

The resulting model is similar to that of a blocking job shop problem, a block section corresponding to a blocking machine, and a train corresponding to a job. The resulting model of the train scheduling problem is similar to that of blocking job shop scheduling problem, a block section corresponding to a blocking machine, and a train corresponding to a job. In our definition of blocking constraint a job (a train), having completed processing on a machine (block section), remains on it until the next machine becomes available for processing. There is a decision variable for each pair (train, block section), corresponding to the time at which the train enter the block section. One difference with the blocking job shop problem is that fast trains require two (or even more) block sections at a time, in order to travel at their maximum speed. Moreover, there are additional constraints concerning departure and arrival times, connections between trains, etc.

4 Formulation of the train scheduling problem

In this section we illustrates some examples of alternative graphs associated with typical constraints arising in train scheduling. In what follows we enumerate, from 1 to n, all the pairs (train, block section), and indicate with $B(i)$ the block section associated with operation i. With this notation, the variables of the problem are the times t_i at which the associated train enter the corresponding block section $B(i)$. For example, the constraint that a train must travel at a minimum speed within a block section corresponds to a maximum travel time δ_i for the train within a given block section, i.e. to a maximum time allowed for completing the associated operation i, and starting the subsequent operation $\sigma(i)$. Let us call p_i the processing time of operation i, i.e. the travel time associated to the i-th pair (train, block section). Figure 1 gives a graphical representation of this constraint.

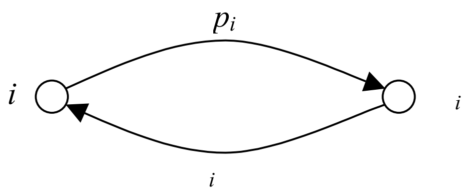

Figure 1: Minimum travel time constraint

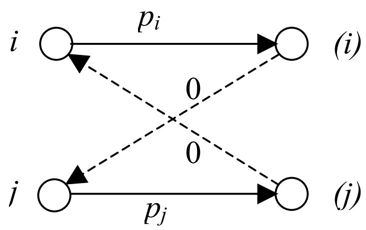

Figure 2: Blocking constraint

We next represent the blocking constraint with a pair of alternative arcs. Recalling the definition of blocking operation, let us now consider two operations i and j, such that $B(i) = B(j)$. Since i and j cannot be executed at the same time, we associate with them a pair of alternative arcs. Each arc represents the fact that one operation must be processed before the other. If i is processed before j, $B(i)$ can host j only after the starting time of the subsequent operation $\sigma(i)$, when i leaves $B(i)$. Hence, we represent this situation with the alternative arc $(\sigma(i), j)$. Similarly, If j is processed before i, $B(j)$ can host i only after the starting time of $\sigma(j)$, see Figure 2.

Figure 3 shows an example with two trains moving in the same direction: train A is slow and train B is fast, nodes i and j refer to the same block section k. Here, p_{hk} is the cover time for train h and block section k. If train B precedes A on block section k, train A must wait until the section is empty, i.e. until train B enters section $k+1$. On the contrary, if train A enters block section k before B, then train B must wait until the next two sections are empty, i.e. until train A reaches block section $k+2$.

Clearly, different trains may have different further requirements, such as the maximum departure time from the stop stations, etc. Most of these constraints can be easily modeled by means of the alternative graph. Figure 4 shows a small railway network with 6 block sections (denoted as 1, 6, 7, 8, 9, and 10), a simple station with two platforms (denoted as 3 and 4), and two special resources (denoted as 2 and

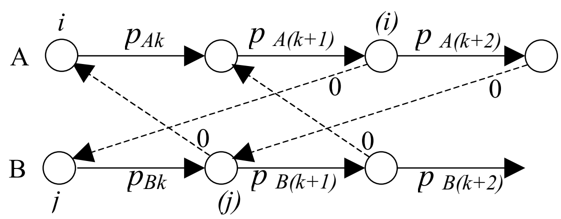

Figure 3: The graph representation for two trains

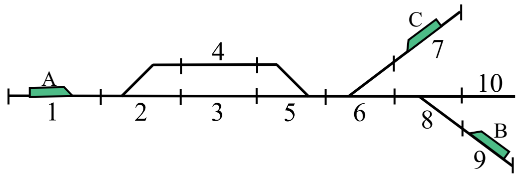

Figure 4: A rail network with three train

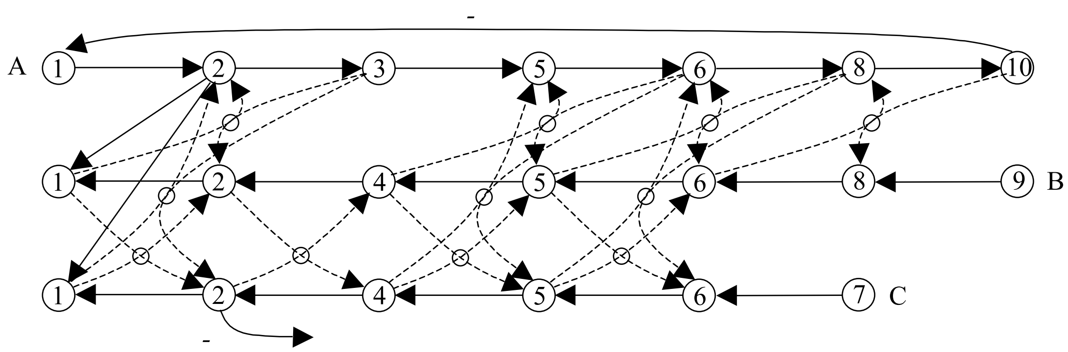

Figure 5: The alternative graph of a rail network with three train

5) that are composed by a set of track segments (called track circuits), and that allow a train to enter or to leave a certain platform. Also these resources have capacity one. Assume that at time t there are three slow trains in the network. Train A is a freight train, going from block section 1 to block section 10, and passing through platform 3 without stopping. Here, α is the time needed to pass through all block sections at the lowest speed allowed. Train B is a passenger train going from block section 9 to block section 1, and passing through platform 4. Finally, train C is a passenger train going from block section 7 to block section 1, and stopping on platform 4. Its latest departure time from the station is β.

Figure 5 illustrates the alternative graph for this small train scheduling problem. For sake of clarity we indicate each node of the alternative graph by the pair (train, block section). Each alternative pair of arcs is associated to the usage of some common resource. In particular, trains A and B share resources 1, 2, 5, 6, and 8. Trains A and C share resources 1, 2, 5, and 6. Trains B and C share resources 1, 2, 4, 5, and 6. Note that the initial position of train A implies that B and C are not allowed to precede A on block section 1, and therefore we have the selected alternative arcs (A2,B1) and (A2,C1). The respective forbidden alternative arcs are not depicted. The fixed arcs with negative weight represents the maximum travel time constraint for train A, and the due date constraint for C.

5 Tabu Search and Alternative Graph

The Tabu Search is a deterministic local search framework introduced by F. Glover [3], [4]. The basic concept of a local search approach is to define a neighborhood function as a function that for each solution associates a set N of neighbor solutions [1]. At each step the tabu search selects a neighbor in the legal neighborhood, and moves the current solution in the selected neighbor. A legal neighbor is a neighbor that does not belong to the tabu list. The Tabu List is a short memory structure used to avoid cycling and to escape form local optima. However a neighbor on the tabu list can be selected as a move if it is *good* enough, i.e. satisfies an aspiration level.

Let us focus now on some details of the tabu search algorithm for alternative graphs. We can proof that all the alternative pairs that do not belong to the critical path cannot improve the makespan. The rule defining the neighborhood is the following: "invert an alternative pair belonging to the critical path". More precisely, given a complete selection S, the set of arcs P in a critical path of $G(S)$, and an alternative pair $((i,j),(h,k))$ such that $(i,j) \in \{(u,v) : (u,v) \in S \cap P\}$, we define a neighbor of S the new selection $S' = S \setminus \{(i,j)\} \cup \{(h,k)\}$. Note that for general alternative graphs the inversion of an alternative pair on the critical path can introduce positive length cycles in the resulting graph.

When designing a tabu search algorithm, we have two choices: we can try to define (1) a neighborhood that can not contain positive length cycles, or (2) a neighborhood that contains positive length cycles. In the first case in a simple neighborhood that discards positive length cycles the weakly optimal connected property does not hold, i.e. it may happens that the optimal solution can not be reached by every starting point. In the latter case there is the problem of managing unfeasible solutions in order to obtain better results.

In our procedure we chose the latter case, redefining the move in this way: "if the current solution is feasible invert an alternative pair belonging to the critical path, else if the current solution is unfeasible invert an alternative pair belonging to a positive weight cycle". In particular, when $G(S)$ is unfeasible we define a neighbor S' of S as follows. Given the set of alternative arcs belonging to a positive weight cycle C and an alternative pair $((i,j),(h,k))$ such that $(i,j) \in \{(u,v) : (u,v) \in S \cap C\}$, then $S' = S \setminus \{(i,j)\} \cup \{(h,k)\}$.

Note that the makespan criterion cannot be used to evaluate unfeasible points. Hence, in our implementation, in order to deal with unfeasible solutions, we developed two different short memory structures. The first one is a Tabu List and it is used in the feasible solutions space, whereas the second one is used only in the unfeasible regions to lead the search towards feasible solutions. In the latter phase we use an evaluation criterion different from the makespan.

Another common feature of local search algorithms is the choice of the initial solution, i.e. the starting point of the algorithm. There are several approaches to this choice such as applying a greedy heuristic algorithm, or choosing a random point. Note that every point in the solution space can be used as initial solution, even if it is unfeasible.

Our preliminary computational experience shows that tabu search approach is very promising for solving complex scheduling problems modeled by means of the alternative graph formulation.

References

[1] E. Aarts and J.K. Lenstra (eds). *Local Search in Combinatorial Optimization.* J.Wiley & Sons, Chichester, 1997.

[2] E. Balas. Disjunctive programming. Ann. Discr. Math., 5, 3-51, 1979.

[3] Fred Glover. Tabu Search–Part I. *ORSA J. Comput.*, 1,3:533–549, 1989.

[4] Fred Glover. Tabu Search–Part II. *ORSA J. Comput.*, 2,1:4–32, 1990.

[5] A. Mascis and D. Pacciarelli. Machine scheduling via alternative graphs. Report DIA-46-2000, Dipartimento di Informatica e Automazione, Università Roma Tre, Roma, 2000.

[6] B. Roy and B. Sussmann. Les problèmes d'ordonnancement avec constraintes disjonctives. SEMA, Note D.S., No. 9, Paris, 1964.