Computing Redundant Resources for the RCPSP

Jacques Carlier
Laboratoire HeuDiaSyC UMR CNRS 6599, Centre de recherches de Royallieu,60200 Compiègne, France
carlier@hds.utc.fr

Emmanuel Néron
Laboratoire d'Informatique de l'Université de Tours, E3I – 64 Av Jean Portalis, 37200 Tours, France
neron@univ-tours.fr

Abstract

Several efficient lower bounds for the Resource Constrained Project Scheduling Problem (RCPSP) have been proposed recently. Some of them are based on the notion of redundant resource. In this paper we define a redundant function as an application f: [0, B] $\rightarrow$[0, B'] such that $i_1 + i_2 + \ldots + i_k \leq B$ implies $f(i_1) + f(i_2) + \ldots + f(i_k) \leq B'$. Then we explain that redundant functions are very useful for computing redundant resources. We also describe an algorithm for computing all maximal redundant functions. Computational results are reported.

1. Introduction

In the Resource Constrained Project Scheduling Problem (RCPSP), non preemptive activities requiring renewable resources and subject to precedence constraints have to be scheduled to minimize makespan. RCPSP has been proved to be NP-Hard and because of its practical interest has been studied by numerous authors in the literature, e.g., [1], [3], [6],[8], and [9].

The aim of this paper is to describe an efficient technique for computing redundant resources for the RCPSP. Indeed redundant resources are very useful for solving RCPSP as it has been proved in two recent papers: they can be used directly as linear constraints in a general linear programming formulation to compute lower bounds [5], or they can be used as new resources which are added to the initial RCPSP instance [4], in order to improve the efficiency of satisfiability tests and time-bound adjustments [2]. This technique is based on the notion of redundant function. We define a redundant function as an application f: [0, B] $\rightarrow$[0, B'] such that : $i_1 + i_2 + \ldots + i_k \leq B$ implies $f(i_1) + f(i_2) + \ldots + f(i_k) \leq B'$, where [0, B] and [0, B'] denote intervals of consecutive integers. B is the availability of some particular initial resource of the RCPSP and B', the availability of the redundant resource. If a denotes the demand of some activity of the initial resource, f(a) denotes the demand of the same activity of the redundant resource. Thus we deduce the way to associate a redundant resource with a redundant function. Of course there are numerous redundant functions, and only few of them are useful.

Indeed we can restrict our search to Maximal Redundant Functions (MRF). We say that a redundant function f is maximal, if there is no redundant function f' such that f'>f. We state the MRF properties and describe an algorithm for computing all MRF.

The organization of the paper is as follows. In section 2, we report properties of MRF. Section 3 is devoted to our algorithm for computing all MRF. Computational results are given in section 4. Discussion and final remarks are reported in section 5.

2. Definition and properties

The aim of this section is to define formally redundant functions and maximal redundant functions (MRF), and to state some fundamental properties of MRF.

Definition 1: A Redundant Function is a discrete application of [0, B] in [0, B'] ($B \in \mathbb{N}$, $B' \in \mathbb{N}$) such that ($i_1 + i_2 + \ldots + i_k \leq B$) implies ($f(i_1) + f(i_2) + \ldots + f(i_k) \leq B'$).

Definition 2: A Maximal Redundant Function (MRF) is a redundant function such that it does not exist any redundant function f' with f'> f, i.e., $\forall i \in [0,B] : f'(i) \geq f(i)$ and $\exists j \in [0, B] : f'(j) > f(j)$.

Property 1: A MRF is an increasing function.
Proof: Indeed, if f is a redundant function, f' defined by $f'(i)=\max_{j \leq i} f(j)$ remains a redundant function. Moreover f'≥f. ð

Property 2: A MRF is such that $f(i) \geq \max_{i=i1+i2} (f(i_1) + f(i_2))$.
Proof: Indeed, if f is a redundant function, f' defined by $f'(i) = \max_{i=i1+i2} (f(i_1) + f(i_2))$, remains a redundant function. Moreover f'≥ f. ð

Theorem 1: Let f be a function of [0, B] →[0, B']. f is a MRF if and only if :

(1) f is increasing,
(2) $f(i) \geq \max_{i=i1+i2} (f(i_1) + f(i_2))$,
(3) $i_1 + i_2 = B$ and $i_1 \neq i_2$ implies that $f(i_2) = B' - f(i_1)$,
(4) $i_1 + i_2 = B$ and $i_1 = i_2$ implies that $f(i_1)= B'/2$ if B' is even, $f(i_1) =(B'-1)/2$ if B' is odd.

Proof: Let $i_0 = (B/2) -1$ if B is even, and $i_0 = (B-1)/2$ if B is odd. If f is a MRF, f satisfies (1) and (2). Let us define f' by $f'(i) = f(i)$ ($i= 0, \ldots, i_0$) and $f'(i) = B' - f(B-i)$ for $i > i_0$ and $i \neq B - i$. It is obvious that f' ≥ f. Moreover f' > f, if f does not satisfy (3) and (4). Consequently, if f satisfies (2) it is a redundant function. It is maximal because of (3) and (4).
Conversely, let f satisfying (1) (2) (3) and (4). Because of (2) f(0)=0. Then (3) implies f(B)=B'. If $i_1+i_2+\ldots+i_k = h \leq B$, (2) implies $f(i_1)+f(i_2)+\ldots f(i_k) \leq f(h)$. f increasing implie: $f(h) \leq f(B)$.
So f is a redundant fucntion. It is a MRF because (3) and (4).ð

3. An algorithm for computing all MRF

Theorem 1 characterizes MRF. So, to compute all of them, we have to enumerate all applications of [0, B] in [0, B'] satisfying (1)(2)(3) and (4). Algorithm ENUMERATION below performs such an enumeration.

Algorithm ENUMERATION(B, B')

(a) Initialisation
  $f(0) = f(1)=\ldots= f(i_0) = 0$;
  $i = i_0$;
(b) Bounding of f
  $LB(j) = f(j)$; $j = 0,\ldots, i$
  $LB(j) = \max_{1 \leq k \leq j} (LB(j-k) + LB(k))$; $j = i+1, \ldots B$
(c)Feasibility test
  if $LB(B) > B'$ goto (e);
(d) Computing f
  $f(j) = LB(j)$; $j = 0,\ldots, i_0$
  if B is even $f(i_0+1) = \min (f(i_0+2), \lceil B'/2 \rceil)$;
  $f(j) = B' - f(B-j)$; $j = i_0+1, .., B$
  $i= i_0+1$;
  Store f *as a new MRF;*
(e) Backtrack
  $i = i -1$; $f(i) = f(i)+ 1$;
  If ($i \neq 0$), goto (b);

Theorem 2 : Algorithm ENUMERATION builds all MRF in $O(B^2 ||MRF||)$, where ||MRF|| is the number of MRF with B and B' set.

Proof : Indeed it goes through an enumeration tree by a Depth First Search. At level i of the tree all the possible values of f(i) are considered. When f(1)…f(i) are set, one can easily verify if it exists at least a MRF with these values, due to theorem 1. Indeed LB(i+1), LB(i+2), …, $LB(i_0)$ are the smallest values for such a MRF. They can be easily completed by using (3) and (4). So we can go directly to the leftest pendant node of the tree. Next it is necessary to backtrack in the search tree until we get some new MRF. It justifies algorithm ENUMERATION. ð

4. Computational Results

In this section we present some experiments to compare the MRF, with the classical redundant resources that have been generated previously [4]. Our main motivation is to estimate the number of MRF for a given couple (B, B'). The following tables present some results concerning the number of MRF for fixed values of B and B'. Table 1 presents these results for small values of B and B', whereas Figure 2 presents the number of MRF for some values of B and some small values of B'. Figure 3 presents the total number of MRF with B' ≤ B, for large values of B. For B fixed and two values of B', the same MRF can be generated. We take care to generate a MRF only once. In the three tables below, we only take into account the MRF generated for the smallest values of B', which is a-priori the most interesting one.

B = 2	B = 3	B = 4	B = 5	B = 6	B = 7	B = 8	B = 9	
1	1	1	1	1	1	1	1	B' = 1
1	0	1	1	1	2	2	1	B' = 2
	2	1	1	2	3	2	2	B' = 3
		1	1	1	4	2	3	B' = 4
			3	2	4	4	4	B' = 5
				3	6	3	3	B' = 6
					7	5	6	B' = 7
						4	4	B' = 8
							8	B' = 9

Table 1 : number of MRF for fixed (B, B')

	B=20	B=25	B=30	B=90
B' = 2	4	4	5	15
B' = 3	5	6	8	23
B' = 4	5	7	8	26
B' = 5	10	12	14	44

Table 2 : number of MRF for large values of B, and small values of B'

B	2	5	10	11	20	30	40	50	60	70	80	90
Total Nb. MRF	2	7	48	66	393	1379	3410	6870	11584	19087	28411	40634

Table 3 : total number of MRF for large values of B,

We want to point out the fact that this approach can be useful for small values of B'. Then the number of MRF to take into account is not large, so they can be directly used as new linear lower bounds in the linear programming formulation that have been proposed [5]. In the latter case it also

may be interesting to generate the corresponding redundant resources to get both satisfiability tests and time-bound adjustments [2], [4].

By using ENUMERATION(B, B'), we have generated more MRF than the initial ones that have been already determined in [4]. Some of those determined by using ENUMERATION(B, B') are linear combination of the initial ones. A theoretical study to prove that ENUMERATION(B, B') provides MRF that cannot been deduced from the initial ones should be done. Moreover our new approach is quite fast and very generic. So it can be applied for very large values of B.

5. Conclusion and Discussion

We have proposed a theoretical and practical extension of redundant resources. These redundant resources can be used both to compute efficient lower bounds or directly to perform some adjustements on release dates and deadlines of activities.

It remains some theoretical open questions : is there any interest to consider MRF verifying B'> B and is there any property to ensure that a MRF can be useful to get efficient lower bounds? Moreover the algorithm that we have presented processes a basic enumeration of the MRF. It can be better to try to search directly the "best"ones.

Finally, the practical impact of these MRF on lower bound computation must be evaluated on many kinds of representative RCPSP instances: KSD instances [7], which are more or less disjunctive, BL instances which are highly cumulative [2], and some generated instances [4].

References

[1] Baar T., Brucker P. and Knust S., Tabu search algorithms and lower bounds for the Resource-Constrained Project Scheduling Problem. in: Voss S., Martello S., Osman I., Roucairol C. (eds): Meta-heuristics: Advances and Trends in Local Search Paradigms for Optimization, Kluwer.

[2] Baptiste Ph., Le Pape C. and Nuijten W., Satisfiability Tests and Time-Bound Adjustments for Cumulative Scheduling Problems, Annals of Operations Research 92(1999)305-333.

[3] Blazewicz J., Cellary W., Weglarz. J and Slowinsky R., Scheduling under Resource Constraints - Deterministic Models. Annals of Operations Research 7, JC Baltzer AG (1986).

[4] Carlier J. and Néron E., A New L.P. Bound for the Cumulative Scheduling Problem, European Journal of Operational Research.127-2(2000)363-382.

[5] Carlier J. and Néron E, On linear lower bounds for the Resource Constrained Project Scheduling Problem. Submitted to European Journal of Operational Research.

[6] Demeulemeester E. and Herroelen W., New Benchmark Results for the Resource-Constrained Project Scheduling Problem, Management Science 43(1997)1485-1492.

[7] Kolish R., Sprecher A. and Drexl A., Characterization and Generation of a General Class of Resource-Constrained Project Scheduling Problem. Management Science, 41 (1995)10,1693-1703.

[8] Mingozzi A., Maniezzo V., Ricciardelli S. and Bianco L., An Exact Algorithm for Project Scheduling with Resource Constraints based on New Mathematical Formulations, Management Science , 44(1999)714-729.

[9] Mohring R.H., Schulz A.S., Stork F. and Uetz M., Resource-Constrained Project Scheduling: Computing Lower Bounds by Solving Minimum Cut Problems, In Nesetril, (ed)., Proc of the 7th annual European Symposium on Algorithms (ESA99), Lecture Notes in Computer Science, Springer Berlin, 1643(1999)139-150.