Decision Making in Manufacturing and Services
Vol. 7 • 2013 • No. 1–2 • pp. 51–68

A Two-Phase Algorithm for a Resource Constrained Project Scheduling Problem with Discounted Cash Flows

Marcin Klimek*, Piotr Łebkowski**

Abstract. This paper presents a Resource-Constrained Project Scheduling Problem (RCPSP) settled by contractual milestones. The criterion analysed here is the maximisation of aggregate discounted cash flows from the contractor's perspective, known as an RCPSP problem with Discounted Cash Flows (RCPSPDCF). The cash flows analysed here cover the contractor's cash outflows (negative cash flows), related to the commencement of individual activities, and cash inflows (positive cash flows) after the fulfilment of individual milestones. The authors propose a two-phase algorithm for solving the problem defined. In the first phase, the simulated annealing metaheuristics is used, designed to identify a forward schedule with as high total DCF as possible. In the second phase, the best first-phase schedule is improved by right shifts of activities. To this end, the procedure which iteratively shifts tasks by one unit is applied, with a view to maximising the objective function. Activity shifts take into consideration precedence and resource constraints, and they are performed for a specified resource allocation to activities. This paper also includes an analysis of the problem for a sample project. The results of computational experiments are then analysed. The experiments were run with the use of standard test problems from the Project Scheduling Problem LIBrary (PSPLIB), with additionally defined cash flows and contractual milestones.

Keywords: resource-constrained project scheduling, discounted cash flows, milestones, heuristics

Mathematics Subject Classification: 90B35; 90C59; 90C35

Revised: December 19, 2013

1. INTRODUCTION

In recent years, the number of structural, construction, IT, and production orders to be executed as projects has been growing. In response to practical demand, numerous research projects have been carried out relating to project scheduling. Various optimisation models and schedule assessment criteria have been analysed. From a practical

* Pope John Paul II State School of Higher Vocational Education in Biala Podlaska, Poland. e-mail: marcin_kli@interia.pl
** AGH University of Science and Technology, Faculty of Management, Department of Operations Research and Information Technology, Krakow, Poland. e-mail: plebkows@zarz.agh.edu.pl

point of view, what is especially important is to schedule optimisation, taking into account economic criteria, that is DCF maximisation.

While scheduling a project with cash flow optimisation criterion, the scheduling results include not only the activity start times, but also the projection of the related cash flows. The financial aspects are most often included in the analysis by taking into consideration the change of money value in time, with the use of dynamic (discount) methods. The Net Present Value (NPV) of future cash flows is computed, taking into consideration the discount rate used to compute interest on the principal amount. Russell (1970) was the first to develop a model (the Max-NPV model) in which the discounted cash flows generated during project execution were optimised. NPV is the most common economic assessment criterion for project scheduling (Hartmann & Briskorn, 2012).

Under an NPV maximisation problem, the following are considered from the contractor's perspective: cash inflows: payments made to the contractor by the client and cash outflows: expenditures representing payments made by the contractor to the client (e.g. liquidated damages), suppliers, employees etc. Expenditures are, *inter alia*, connected with the execution of works and the consumption of resources. Inflows include e.g. the payments for the completion of individual sections of the project. The contractor's outflows are usually more frequent than inflows, and the value of outflows depends on the costs incurred.

The aggregate discounted cash flows are affected by a number of factors, including the client's schedule of payments to the contractor. In current research (Mika *et al.*, 2005; Ulusoy *et al.*, 2001), various Payment Project Scheduling (PPS) models are considered. In such models, such client's schedule of payments is sought which will allow to maximise the project's NPV. The parameters determined in the PPS optimisation process include: the client's total payment under the project, the number of payment instalments, as well as the amounts and payment dates of subsequent instalments. The considered PPS models include the following four (Mika *et al.*, 2005; Ulusoy *et al.*, 2001):

– Lump-Sum Payment (LSP): one-off payment by the client after the project completion;
– Payments at Event Occurrences (PEO): events refer to work completion, e.g. payments are made upon the completion of agreed works or after each work, known as Payments at Activities' Completion times (PAC);
– Equal Time Intervals (ETI): payments are made in equal agreed periods, with the agreed number of payments;
– Progress Payments (PP): payments at time intervals, with an unspecified number of payments.

While scheduling cash flows, the following intuitive principle is observed: inflows (positive cash flows) should be obtained as soon as possible, while outflows (negative cash flows) should be effected as late as possible. Bonus-penalty systems are also examined as payment models (He & Xu, 2008). For example, the penalties for delays of the contractual terms of work execution, while bonuses – for early completion. Time windows (time intervals) are assigned to the activities. Activity com-pletion is neither awarded, nor punished within such periods. The contractor either receives a bonus for early activity (milestone) completion, or is charged with a penalty for a delay. A review of project scheduling research taking into account economic criteria and cash flows can be found in analytical papers (Hartmann & Briskorn, 2012; Herroelen *et al.*, 1997; Kolisch & Padman, 2001; Węglarz, 1999). Those papers also address other issues which are not discussed here.

In this paper, a Resource-Constrained Project Scheduling Problem (RCPSP), with predefined milestones, is examined. Schedule assessment uses economic criteria, namely, maximising aggregate cash flows RCPSPDCF, NPV, with a system of penalties (in the form of reduced payments to the contractor by the client, in the case of delays in milestone attainment), and bonuses (in the form of early payments to the contractor by the client, in the case of early milestone attainment). This is an optimisation model developed by the respective authors.

RCPSP is an NP-hard problem, being a generalisation of a classical NP-hard Job Shop problem (Błażewicz *et al.*, 1983). Consequently, in the case of the projects with a large number of activities, it is justified to use approximate algorithms, that is priority algorithms, insert algorithms, or metaheuristics algorithms. In this paper, the authors use one of the effective metaheuristics: the Simulated Annealing (SA) algorithm. The solution found by the SA algorithm and the known Schedule Generation Schemes (SGS) are not, as a rule, optimised in respect of the problem of DCF maximisation considered here. Schedule improvement by shifting activities right is possible. Work execution commencement is associated with expenditures and from the contractor's perspective, it pays off to postpone the expenditures. On the other hand, such a postponement may also postpone milestone attainment, thus reducing the client's discounted payments. It would be advisable to find a procedure to verify which shifts increase the project's aggregate DCF's.

In the proposed two-phase algorithm, the best schedule found by the SA algorithm in the first phase is improved in the second phase by right shifts of the activities, with the use of a procedure which iteratively performs unit shifts of activities, driven by the criterion of ob-jective function maximisation. In each iteration, only the activity whose shift adds most to the project NPV is shifted right. Activity shifting takes into consideration precedence and re-source constraints, as it is performed for specific resource allocation to activities.

The purpose of this paper is to present a model of staged project settlements and a new project NPV maximisation algorithm, as well as prove the applicability of this algorithm to the problem under consideration. Also, the authors present an example describing the problem and the algorithm. Finally, the results of computational experiments are analysed for test activities sourced from the PSPLIB library (Kolisch & Sprecher, 1997), with additionally defined project cash flows and milestones.

2. PROBLEM FORMULATION

A project is a collection of interrelated activities (tasks) executed with the use of resources, with a view to achieve assumed objectives. The project scheduling problem most often boils down to the determination of activity commencement times for the adopted optimising criterion. A project is modelled as a $G(V,E)$ graph, with Activity On Node (AON). Table 1 lists notations applied for the project scheduling problem definition, with predefined agreed milestones, whose attainment triggers partial payments under the project.

Table 1. *Parameters and variables – symbols*

n –	number of activities in the project,
i –	activity number,
$G(V,E)$ –	acyclic digraph, modelling the project in the AON representation,
V –	set of nodes representing individual activities,
E –	set of edges (arcs) representing ordering relations between activities,
k –	resource type number,
K –	number of resource types,
a_k –	number of available resources of type k,
$A(t)$ –	set of activities executed in a time interval $[t-1,\ t]$,
d_i –	duration of activity i,
r_{ik} –	demand for type k resources during the execution of activity i,
ST_i –	start time of activity i planned in the current schedule,
FT_i –	end time of activity i planned in the current schedule,
M –	number of milestones,
m –	milestone number,
mt_m –	contractual completion time of milestone m,
MT_m –	completion time of milestone m planned in the current schedule,
MA_m –	set of activities executed in milestone m,
α –	discount rate,
CFA_i –	the contractor's expenses for the execution of activity i, incurred at its start,
PM_m –	contractual amount of the client's payment for the execution of milestone m,
CM_m –	contractual unit cost of a penalty for a delayed execution of milestone m,
CFM_m –	the client's payments for the execution of milestone m, as computed in the current schedule,
ER –	set of additional edges (arcs): pairs of activities (i,j) which are not sequentially related in the original activity network $G(V,E)$, but between which resource flows occur, i.e. $f(i,\ j,\ k)\ >\ 0$,
$f(i,j,k)$ –	for each resource k, the number of such resources transferred from the end of activity i to the start of activity j.

The optimisation criterion that is most often used for the RCPSP problem is the time-related criterion: makespan minimisation. Other frequently used criteria include DCF optimisation (RCPSPDCF) and project NPV maximisation (the most often used economic criterion). This paper analyzes the authors' model of the RCPSDCF problem, providing for the maximisation of discounted cash flows connected with the execution of activities and milestones. The model may be described with the following Formulas (1–5).

Maximisation of F:

$$F = \sum_{i=1}^{n} (CFA_i \cdot e^{-\alpha \cdot ST_i}) + \sum_{m=1}^{M} (CFM_m \cdot e^{-\alpha \cdot MT_m}) \tag{1}$$

in the presence of the following conditions and constraints:

$$\forall (i,j) \in E : ST_i + d_i \leqslant ST_j \tag{2}$$

$$\forall t, \forall k : \sum_{i \in A(t)} r_{ik} \leqslant a_k \tag{3}$$

$$\forall i \in MA_m, tm_m < tm_{m+1}, m \in< 1, M) : FT_i \leqslant tm_m \tag{4}$$

$$CFM_m = PM_m - CM_m \cdot \max(MT_m - tm_m, 0) \tag{5}$$

The objective of scheduling is to identify the vector of activity of start times ST_i, with the maximum value of the objective function F (see Formula 1), taking into consideration precedence constraints (see Formula 2) and resource constraints (see Formula 3). A non-preemptive, single mode RCPSP problem is considered.

The schedule sought for should take into consideration the agreed milestones (see Formulas 4–5). If the execution of any milestone is delayed, the schedule is executable, but the payment by the client is reduced (see Formula 5), which in turn reduces the aggregate discounted cash flows (see Formula 1). Milestone execution before the contractual deadline brings benefits, as the payment by the client is also made earlier than expected, and that increases the discounted value of the payment. The cash flow maximisation problem is analysed from the project contractor's perspective. For the contractor, inflows are the client's payments for milestone attainment: CFM_m. Expenses (CFA_i) are related to activity completion. It is assumed here that the contractor incurs expenses CFA_i (i.e. the costs of using resources and materials for the execution of activity i at the time of activity start, as planned in the nominal schedule, and the contractor receives inflows CFM_m just at the milestone's end time. Client payments CFM_m (see Formula 5) for the execution of milestone m are contractual amounts PM_m reduced by the amounts charged for the delayed execution of milestone m. All cash flows are discounted at rate α.

For the client, it is not profitable to spend money earlier. However, the client may be interested in the proposed settlement system. Settlement by milestones enables the client to control the project's progress throughout its makespan; additionally, a contractual penalty system "encourages" the contractor to accelerate activity completion.

There is a conflict of interests between the client and the contractor when it comes to the dates and amounts of cash flows. It is those differing expectations of the client and of the contractor that make the developed bonus-penalty-system model useful. Model parameters should be selected with a view to stimulating the contractor to execute project activities as soon as possible. From the contractor's perspective, the benefits from the client's earlier payments for attained milestones should exceed the contractor's costs related to accelerating activity execution, while a penalty for a delayed execution should exceed the contractor's benefits from delaying activity (milestone) completion. From the client's perspective, a bonus owed to the contractor for early execution should not exceed the client's benefits from such early execution, while a penalty for a delayed execution should be higher than the client's profit lost due to such a delay.

3. TWO-PHASE ALGORITHM OF PROJECT'S NPV MAXIMISATION

In the model analyzed (see Formulas 1–5), cash inflows are the client's payments for the attained milestones and cash outflows are the contractor's expenses incurred for activity execution (upon work commencement). Milestone events based on a payment approach is studied in the literature, e.g. (He *et al.*, 2009; He *et al.*, 2012), but for different models and problems. Current research on project scheduling with NPV maximisation has not covered the so defined a problem. The problems covered rather include the assignment of cash outflows and/or inflows to individual activities (Baroum & Patterson, 1996; Icmeli & Erenguc, 1996; Pinder & Marucheck, 1996; Mika *et al.*, 2005; Selle & Zimmermann, 2003; Ulusoy & Özdamar, 1995; Vanhoucke *et al.*, 2001; Vanhoucke, 2006). For such problems, algorithms are sought for that would maximise the project's NPV by scheduling activities, with cash inflows as soon as possible, and those with cash outflows as late as possible.

For optimisation problems, it is important to identify a convenient and effective representation (coding) for potential solutions. For the RCPSP problem, the most effective (Hartmann & Kolisch, 2000; Kolisch & Padman, 2001) is the solution coding, used also here, in which the problem is coded as a list of activities, that is a permutation setoff tasks $1, 2, \ldots, n$, taking into consideration precedence constraints. Based on the activity list, with the use of decoding procedures known as Schedule Generation Schemes (SGS), a schedule is defined, that is, most often, the start times of individual activities. In this paper, forward scheduling procedures are used, designed to determine as early individual activity start time as possible, subject to the precedence relations and resource constraints.

In the SA algorithm developed here, the serial SGS and the parallel SGS procedures (Kolisch, 1996) are considered as forward scheduling decoding procedures. In the serial SGS, in each consecutive iteration, the earliest possible time is identified for the first unscheduled activity in the activity list (subject to precedence relations and resource constraints). In the parallel SGS, iteratively, in consecutive time instances t, all those unscheduled activities are started (analyzed in the order of their appearance on the activity list) whose execution is feasible at that time (subject to precedence relations and resource constraints).

For the DCF maximisation problem, solutions generated by SGS procedures admit improvement by postponing (shifting right) the activities with cash outflows and/or shifting left the activities with cash inflows. Various solution improvement procedures by shifts are considered, however, they are not feasible in the case of problems providing for project settlement by milestones. The review of these procedures is included in the paper (Vanhoucke, 2006).

In this paper, a two-phase algorithm is proposed, adapted to the problem under consideration. In the first phase, a schedule is created without right shifts of activities. An approximate algorithm, SA metaheuristics, is used in this phase. The effectiveness of the SA metaheuristics has been proved in the research on the RCPSP problem, against various criteria, including project NPV maximisation or makespan minimisation (Hartmann & Kolisch, 2000; Bouleimen & Lecocq, 2003; Boctor, 1996). In the second phase, the best solution identified in the first phase is improved, with iterative unit right shifts of activities, with a view to maximising objective function F. Figure 1 below presents the operation of the proposed two-phase algorithm.

```
Initialize x, T0, $\lambda$,  $T_k$            //start of first stage
$T$ := $T0$;
$x*$ := $x$;
repeat
  select y - neighbour solution for x
  if(f(y)< f(x*)) then
    x* := y;
  if(P(f(y),f(x),T) > RND) then
    x := y;
  actualize temperature T
until(termination criteria not met);      //end of first stage
S* := SGS(x*);
resourceallocation(S*);                   //start of second stage
repeat
  bestF := F(S*)
  for i := 1 to n do
    begin
    S' := rightshift(S*, i);
    if (F(S') > bestF) then
    begin
      bestActivity := i;
      bestF := F(S');
    end
  end
  if (F(S*) < bestF) then
   S* := rightshift(S*, bestActivity);
until (F(S*) < bestF);                     //end of second stage
return S*
```

Where: T is the current SA temperature, T_0 is the initial (maximum) temperature, T_k is the final (minimum temperature), λ is the lambda parameter, RND is a random number from uniform distribution on $(0,1)$, $x*$ is the best current solution (activity list), x is the current solution, $f(x)$ is the value of the objective function SA for the solution x : $f(x) = -F(SGS(x)), SGS(x)$ is the method generating a schedule, with the use of the SGS procedure applied to the activity list x, $F(S)$ is the value of objective function F of the problem analyzed for the schedule S, y is the solution obtained by shifting the solution x, 4P(f(y), f(x),T)$istheacceptancefunctionforthesolutiony$,S*4 is the best schedule at the given step, *resourceallocation*$(S*)$ is a procedure generating resource allocation to activities in the schedule $S*$, S' is the schedule analysed at a given time, *bestF* is, in a given iteration, the highest determined value of the objective function, for schedules with various right shifts, *bestActivity* is the number of the activity whose right shift adds most to the value of the objective function F and *rightshift*$(S*, i)$ is a method generating a new schedule from schedule $S*$ by a right shift of individual activity i.

Fig. 1. *Two-phase algorithm for DCF maximisation problem, for a project settled by milestones*

A detailed description of both phases of the algorithm is included in Subsections 3.1 and 3.2.

3.1. PHASE 1: SIMULATED ANNEALING ALGORITHM

The simulated annealing algorithm imitates the process of annealing of solids used in metallurgy. Throughout the optimisation process, the SA algorithm tends to the energy minimum. The characteristic feature of the SA algorithm is the feasibility of leaving local extreme by using a suitable acceptance function, which allows for accepting the solutions that are worse than those found.

There are numerous variations of the simulated annealing algorithm, including cooperative (COSA), modular (MSA), and adaptive (ASA) SA algorithms. The SA algorithm presented here is close to the basic algorithm (Kirkpatrick *et al.*, 1983).

At the beginning of the SA algorithm operation, the start (initial) solution x and algorithm parameters are initialized, including the initial temperature $T0$, final (minimum) temperature T_k, the λ parameter. The parameters and initial solution x may be set during the tuning phase. In consecutive runs of SA, moves are performed (such as insert, swap, adjacent swap etc.) resulting in a new solution y, a neighbour of the then existing solution x.

The new solution y is stored (remembered) if it is the best of the solutions found up to that time. Acceptance criterion is verified with a defined function determining the probability of accepting y. For a classical acceptance function (see Formula 6):

- a better (or equal) solution y, that is one for which $(f(x) \geqslant f(y))$, is always accepted, as then the following inequalities hold: $P(f(y), f(x), T) \geqslant 1 > RND$;
- a worse solution y is accepted if the generated random number $0 < RND < 1$ is less than the acceptance probability.

$$P(f(y), f(x), T) = \exp\left(\frac{f(x) - f(y)}{T}\right) \tag{6}$$

The probability of accepting a worse solution is higher at the beginning of the algorithm operation at higher values of current temperature T.

At the end of a given run of the SA algorithm, current temperature T is changed in accordance with the adopted cooling scheme: logarithmic (see Formula 7) or geometric (see Formula 8); if T_p is the temperature in the p-th run, then for the adopted value of parameter λ and the maximum number of SA runs equal to N, the temperature for the next run is:

$$T_{p+1} = \frac{T_p}{1 + \lambda \cdot T_p}, \qquad \text{for } \lambda = \frac{T_0 - T_k}{N \cdot T_0 \cdot T_k} \tag{7}$$

$$T_{p+1} = \lambda \cdot T_p, \qquad \text{for } \lambda = \left(\frac{T_k}{T_0}\right)^{\frac{1}{N}} \tag{8}$$

In the algorithm developed here, a single solution is verified at a given temperature (other papers consider verifying numerous solutions at a given temperature).

SA runs are repeated until the stopping condition is met, that is the predefined number of iterations, verification of a predefined number of solutions, or the decrease in the current temperature up to the final temperature. For the permutation representation, numerous solution space search operators (moves) are known. In the case of the RCPSP problem, a move should keep the precedence relations between activities. The experiments have covered the analysis of the following moves:

– Insert: an activity is selected at random and then moved to a randomly selected position between its last predecessor on the activity list and its first successor;

– Swap: an activity is selected at random and then swapped with a randomly selected activity appearing between the direct predecessor of the former activity on the activity list and its direct successor;

– Adjacent Swap: two adjacent activities in the activity permutation are swapped; an activity is selected at random and then swapped with the next activity on the list, subject to precedence relations.

Experiments run by the authors serve, in particular, the purpose of identifying the most efficient parameters for the simulated annealing algorithm.

3.2. PHASE 2: ITERATIVE PROCEDURE OF UNIT RIGHT SHIFTS OF ACTIVITIES

The improvement algorithm developed by the authors supports right shifts of selected activities in solution S^* obtained with the use of forward scheduling. In consecutive iterations of the algorithm, activities are right shifted by one unit of time (activity execution start time is delayed by one unit of time). In each iteration, right unit shifts of all activities are tested in order to identify this transformation of the schedule for which objective function F takes the largest value. The procedure stops at the iteration in which no right shift improves the value of F. The improvement algorithm proposed here is versatile: it also supports the optimisation of objective functions other than those analysed here.

When applied to the RCPSP problem, shift procedures have to take into consideration the resource constraints (Vanhoucke, 2006). Identification of resource allocation with the $resourceallocation(S^*)$ method facilitates the right shift operation. A given nominal schedule (specified activity start times $ST_1, \ldots, ST_n$) admits numerous different resource allocations, for which the proposed algorithm generates different solutions, using right shifts. The *rightshift*(S^*, i) method generates different schedules S' for different resource allocations, that is it determines different start times for activities starting after the completion of activity i.

The resource allocation problem for the RCPSP problem is strongly NP-hard, even with one resource type (Leus, 2003). Resource flow networks are used to describe the resource allocation problem (Leus, 2003). A resource flow network is composed of the arcs (edges) of the original activity network $G(V, E)$ and additional arcs (forming a set E_R). In a resource flow network, there appears each pair of such nodes (activities) between which resource flow occurs, being such that $f(i, j, k) > 0$.

The constraints for the resource allocation problem are formulated as follows (Deblaere *et al.*, 2006):

- The sum of all resources of a given type outflowing from the dummy start activity 0 equals the sum of those resources inflowing to the dummy finish activity $n+1$ and amounts to a_k (for each resource type k):

$$\forall k \in K : \sum_{j\in V} f(0,j,k) = \sum_{j\in V} f(j,n+1,k) = a_k \tag{9}$$

- The sum of all resources of a given type inflowing to a given node representing non-dummy activity equals the sum of those resources outflowing from that node and amounts to r_{ik} (for each resource type k):

$$V\backslash\{0,n+1\}\forall k \in K : \sum_{j\in V} f(i,j,k) = \sum_{j\in V} f(j,i,k) = r_{ik} \tag{10}$$

Research into the resource allocation problem covers the problem of robust resource allocation, which reduces to the minimisation of the number of additional arcs (Deblaere *et al.*, 2006; Leus, 2003; Policella *et al.*, 2004; Policella, 2005), as each additional arc in E_R is a new precedence constraint, which decreases schedule robustness. With a view to such minimisation, the same resources are allocated to the execution of activities directly connected with each other in schedule ordering (Policella *et al.*, 2004; Policella, 2005), aggregate flows between individual activities are maximised (Deblaere *et al.*, 2006) and/or the effect of potential production interferences is minimised (Deblaere *et al.*, 2006; Leus, 2003). A review of resource allocation algorithms is included in the papers (Deblaere *et al.*, 2006; Klimek & Łebkowski, 2011). In this paper, those procedures are described which have been used in the computational experiments.

Robust resource allocation procedures are applicable to resource allocation for the problem under consideration. Intuitively, the lesser the number of additional, non-technological precedence constraints, the greater the number of permitted activity shifts. Accordingly, it is reasonable to test the known resource allocation procedures (developed for the problem of additional arc number minimisation) for the feasibility of generating solutions by right shifts for the analysed problem of DCF maximisation in a project settled by milestones.

Resource allocation algorithms use the concept of chains and Partial Order Schedules (POS) (Policella *et al.*, 2004; Policella, 2005). In the simplest BasicChaining algorithm, an activity is assigned to the first free chain connected with the next resource. Exercisable resource flow networks are created, without taking into consideration optimisation criteria. The original network of the project is enhanced, often in excess, with new precedence constraints, which render the start time of a given activity dependent on end times of other activities (the cardinality of the set ER is not minimised).

In the case of the Iterative Sampling Heuristic (ISH) algorithm, the number of additional arcs is reduced by assigning activities whose demand for a given resource type exceeds 1, in a way which would maximise the number of common chains with the last activities in the available chains.

The ISH algorithm ignores precedence relations in the original project network $G(V, E)$. In the ISH^2 procedure, each analyzed activity $i = 1 \dots n$ is first assigned to chains whose last activity is the direct (technological) predecessor of activity i. The use of the ISH^2 algorithm reduces the number of additional arcs; it reduces the number of what is known as synchronisation points.

The next algorithm tested for the purposes of this paper is ISH^2-UA (Klimek, 2010). Its operation is similar to that of ISH^2, but here, the first launched procedure is the procedure of identifying unavoidable arcs (Deblaere *et al.*, 2006). Each activity is first assigned to chains whose last activity is the direct predecessor of the given activity or is connected with it by an unavoidable arc.

4. ILLUSTRATIVE EXAMPLE

Let us assume an example of the project consisting of eight non-dummy activities executed with the use of a single resource type of availability 10. The milestones are defined with deadlines 4, 9 and 12.

Figure 2 shows information on the project, its milestones and cash flows (activities of the same milestone are marked with the same colour). For the purposes of computing NPV, the discount rate of $\alpha = 0.01$, where the capitalisation period equals 1, was assumed.

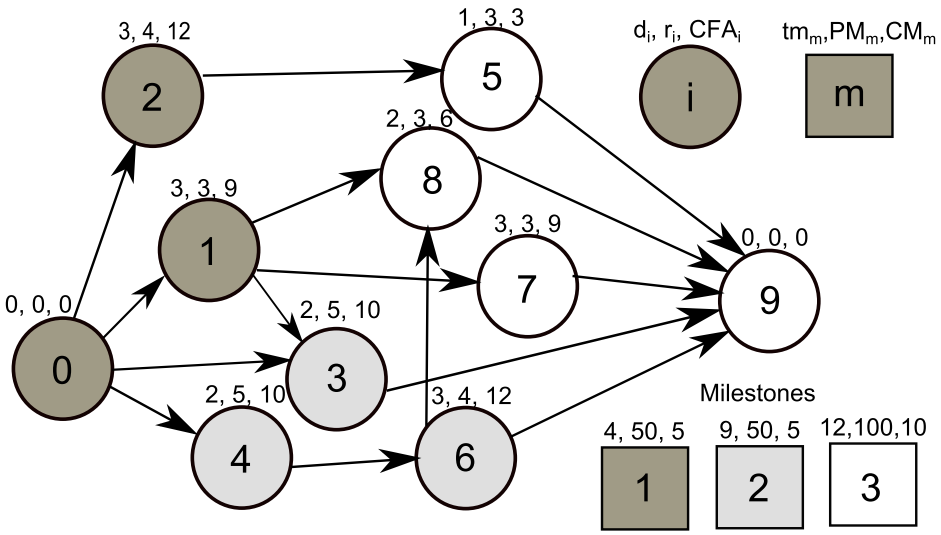

Fig. 2. *AON network for the exemplary project settled by the milestones*

The solutions created in the first phase of the proposed algorithm do not include right shifts. They are generated by known SGS procedures of forward scheduling, used also for other problems, including makespan minimisation. They rearrange the activity list into executable scheduling, taking into consideration precedence and resource constraints.

The second phase of the algorithm deserves a more detailed discussion. In this phase, schedule S will be analysed; it is obtained, for example, with the use of the serial SGS for activity list $\{1, 2, 3, 4, 7, 6, 5, 8\}$, with the value of objective function F equal to 116.38.

Activity start times in schedule S are: $ST_1 = 0$, $ST_2 = 0$, $ST_3 = 3$, $ST_4 = 3$, $ST_5 = 5$, $ST_6 = 5$, $ST_7 = 5$, $ST_8 = 8$ and $ST_9 = 10$. In schedule S, all milestones are completed before the contractual deadlines ($MT_1 = 3$, $MT_2 = 8$ and $MT_3 = 10$, while $tm_1 = 4$, $tm_2 = 9$ and $tm_3 = 12$).

Schedule S admits improvement by unit right shifts of those activities whose postponement will not change milestone end times. In the second phase, solutions are improved by way of iterative unit right shifts of activities. For different resource allocations, the solutions with different values of F (aggregate DCF) are generated.

The resource allocations generated by the BasicChaining, ISH, ISH^2 and ISH^2-UA procedures are shown in Figures 3a, 3d, 3g, and 3j, respectively, while the corresponding resource flow networks are shown in Figures 3c, 3f, 3i, and 3l, respectively. Finally, the revised schedules with activity right shifts are presented in Figures 3b, 3e, 3h, and 3k, respectively.

The use of an improvement algorithm increases the value of the analysed objective function, owing to postponing start times of some activities. In the case of the algorithms shown in Figures 3b and 3e, the value of F is 116.67. The value was increased by six iterative activity shifts: activity 7 was shifted right by 2 time units and, subsequently, activity 5 was shifted right by 4 time units.

The objective function F attains the largest value of 116.86, for the schedules illustrated in Figures 3h and 3k. The second-phase improvement algorithm performed four iterative activity shifts in the following order:

- a right shift of activity 3 (with F growing from 116.38 to 116.59, and forced right shifts of activities 5 and 7 by 1 time unit),
- a right shift of activity 3 (with F growing from 116.59 to 116.80, and forced right shifts of activities 5 and 7 by 1 time unit),
- a right shift of activity 5 (with F growing from 116.80 to 116.83), and finally a right shift of activity 5 (with F growing from 116.83 to 116.86).

The resource flow networks in Figures 3c and 3f include larger numbers of additional arcs than those in Figures 3i and 3l; that is not preferred for the effectiveness of improvement procedure. The schedules in Figures 3b and 3e give the value of objective function F lower than the schedules in Figures 3h and 3k, which is attributable to the presence of the additional arc (3, 6) in the resource flow network.

Accordingly, a right shift of activity 3 delays the start of activity 6, this in turn delays the completion of project milestones 2 and 3, and decreases the value of objective function F.

The example analyzed here indicates that different resource allocations applied to the same schedule can generate schedules with different values of objective function F, using right shifts. It is thus reasonable to run an analysis of known resource allocation algorithms for their efficiency in solving the problem under consideration.

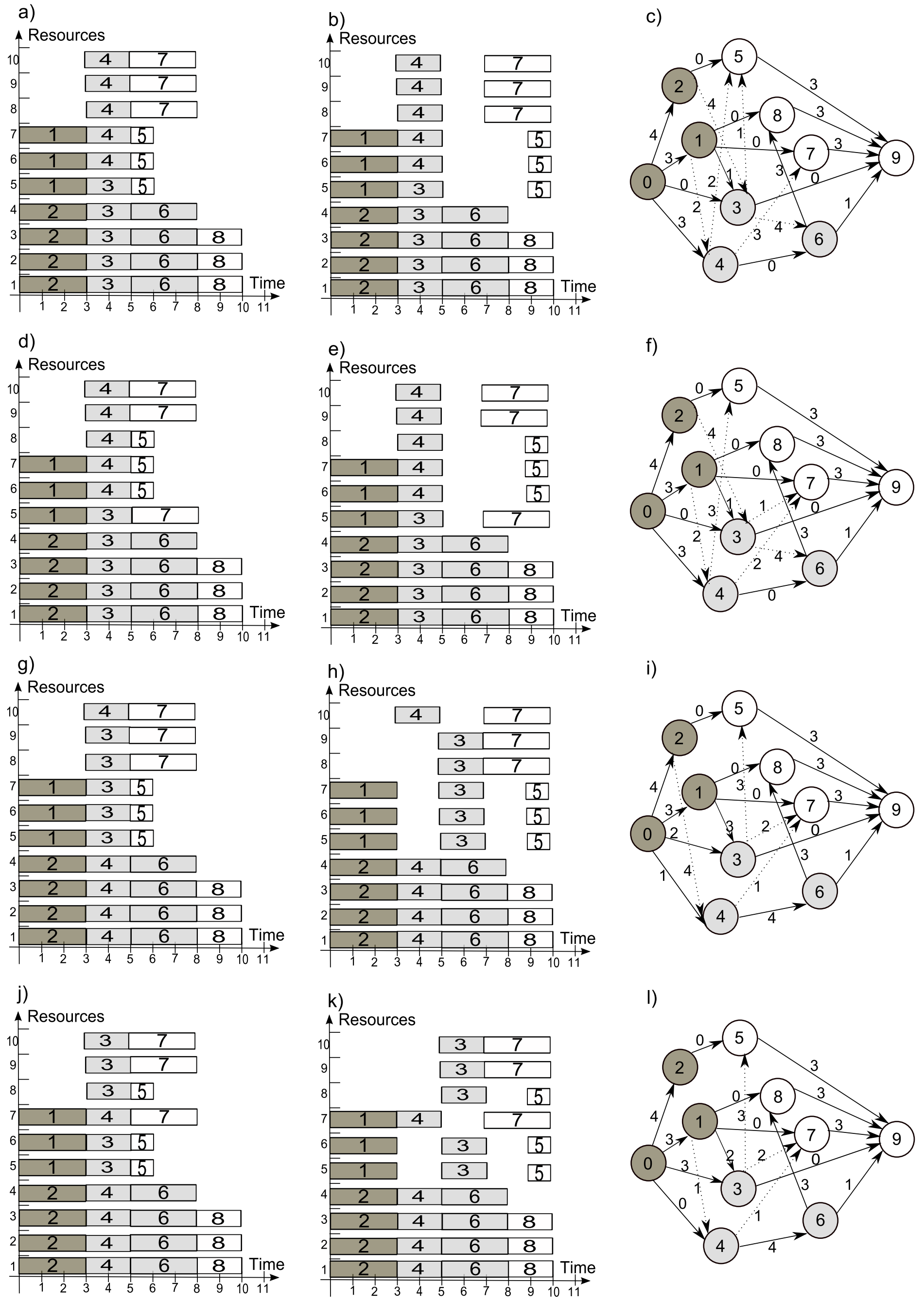

Fig. 3. *Schedule S generated in the first phase with the resource allocations obtained with the use of the BasicChaining (3a), ISH (3d), ISH^2(3g), and ISH^2-UA (3j) procedures, corresponding second-phase schedules (3b, 3e, 3h, and 3k) and resource flow networks (3c, 3g, 3i, and 3l)*

5. THE RESULTS OF THE COMPUTATIONAL EXPERIMENTS

Our experiments have been supported by a program implemented in the C# language, in the Visual Studio.NET environment. 960 test instances were used from sets J30 (30-activity projects) and J90 (90-activity projects), collected from the PSPLIB library (Kolisch & Sprecher, 1997), with additionally defined four milestones generated by the LOSM procedure (Klimek, 2010). For the financial settlement of each project, the following values of parameters were set: $PM_1 = 40$, $PM_2 = 40$, $PM_3 = 40$, $PM_4 = 80$, $CM_1 = 1$, $CM_2 = 1$, $CM_3 = 1$, $CM_4 = 2$, $\alpha = 0.01$. Costs CFA_i (see Formula 7) are calculated pro rata in respect of the demand for resources and duration of a given activity, with the aggregate costs of all activities equal to 100:

$$\forall i \in CFA_i = \frac{d_i \cdot \sum\limits_{k=1}^{K} r_{ik}}{\sum\limits_{j=1}^{n} (d_j \cdot \sum\limits_{k=1}^{K} r_{jk})} \cdot 100, \quad \text{for } \ i = 1 \ldots n \tag{11}$$

The experiments were designed to verify the efficiency of the proposed two-phase algorithm and parameter settings for the algorithm. It's only introduction to extensive experiments which will be performed in further authors' works.

For the simulated annealing algorithm (the first phase of the algorithm in question), experiments used various SGS decoding procedures (serial and parallel ones), various cooling scheme parameters (initial temperatures 5, 1, 0.5 or 0.1, logarithmic or geometric cooling scheme), as well as solution space search techniques (moves: Swap, Adjacent Swap, or Insert). In the SA tuning phase, 100 random schedules are created, with the best of them taken as the start solution. For each of the experiments, the number of solutions examined in the first phase is 1,000. The solution found in the first phase is improved in the second phase by a right-shift algorithm. Different resource allocation procedures are used in the second phase (BasicChaining, ISH, ISH^2 and ISH^2-UA) and their effect on the value of the objective function F is analyzed. For each set of parameters one run of algorithm is performed. The results of applying the two-phase algorithm to projects from sets J30 and J90, for various SA parameter configurations and various resource allocation procedures (in the improvement algorithm) are set forth in Tables 2 and 3.

For the 30-activity projects (set J30), effective is the following algorithm: SA (logarithmic cooling scheme, initial temperature 0.5, move: Insert), with the resource allocation generated by the ISH^2-UA procedure. For the 90-activity projects (set J90), effective is the following algorithm: SA (geometric cooling scheme, initial temperature 0.1, move: Insert), with the resource allocation generated by the ISH^2-UA procedure.

In the first phase, the serial SGS procedure provides better schedules. There is only a minor effect of the cooling scheme on the objective function value; the selection of the initial temperature proves more important. A comparison of local search techniques reveals that the Insert move gives the best solutions. The Swap move produces comparable results. The worst schedules are obtained with the use of the Adjacent Swap move.

The following pattern has emerged: the better the first-phase schedules (with no right shifts), the better the second-phase ones (improved by right shifts). In the second improvement phase, the selection of resource allocation procedure proves a major importance. In our experiments the best solutions are generated for resource flow networks created by the ISH^2 UA procedure, followed by ISH^2, ISH and BasicChaining. The same order of procedures appears for the problems of additional arc number minimisation and robust resource allocation (Klimek, 2010).

Table 2. *Average values of the objective function F for various parameters of the algorithm and for 30-activity projects (set J30)*

Algorithm parameters	Serial SGS					Parallel SGS				
	A1	A2	A3	A4	A5	A1	A2	A3	A4	A5
F1, M1, T1	53.09	53.67	53.72	53.84	53.86	52.56	53.19	53.23	53.35	53.37
F1, M1, T2	53.09	53.67	53.71	53.83	53.85	52.54	53.17	53.21	53.33	53.35
F1, M1, T3	53.04	53.62	53.66	53.78	53.80	52.52	53.15	53.20	53.32	53.34
F1, M1, T4	53.10	53.68	53.73	53.84	53.87	52.54	53.17	53.21	53.33	53.35
F1, M2, T1	52.91	53.51	53.55	53.67	53.69	52.47	53.10	53.14	53.26	53.28
F1, M2, T2	52.87	53.46	53.51	53.63	53.65	52.45	53.08	53.12	53.24	53.26
F1, M2, T3	52.91	53.50	53.54	53.66	53.68	52.41	53.04	53.08	53.21	53.23
F1, M2, T4	52.84	53.44	53.48	53.60	53.62	52.46	53.09	53.14	53.26	53.28
F1, M3, T1	53.08	53.68	53.72	53.84	53.86	52.55	53.17	53.22	53.34	53.36
F1, M3, T2	53.08	53.66	53.71	53.83	53.85	52.55	53.18	53.22	53.34	53.36
F1, M3, T3	**53.12**	**53.70**	**53.75**	**53.86**	**53.88**	52.56	53.18	53.23	53.35	53.37
F1, M3, T4	53.08	53.66	53.71	53.82	53.85	52.54	53.17	53.22	53.34	53.36
F2, M1, T1	52.74	53.35	53.40	53.52	53.55	52.41	53.05	53.10	53.22	53.24
F2, M1, T2	53.01	53.60	53.65	53.77	53.79	52.53	53.16	53.21	53.34	53.36
F2, M1, T3	53.10	53.69	53.74	53.85	53.88	52.55	53.19	53.23	53.35	53.38
F2, M1, T4	53.10	53.68	53.73	53.85	53.87	52.58	53.21	53.25	53.37	53.39
F2, M2, T1	52.67	53.29	53.34	53.46	53.49	52.37	53.00	53.05	53.17	53.19
F2, M2, T2	52.89	53.50	53.54	53.66	53.68	52.44	53.08	53.13	53.25	53.27
F2, M2, T3	52.91	53.52	53.56	53.68	53.71	52.45	53.09	53.13	53.26	53.28
F2, M2, T4	52.93	53.53	53.57	53.69	53.71	52.48	53.11	53.16	53.28	53.30
F2, M3, T1	52.70	53.31	53.35	53.48	53.50	52.36	52.99	53.04	53.16	53.18
F2, M3, T2	52.93	53.53	53.58	53.70	53.72	52.48	53.11	53.15	53.28	53.30
F2, M3, T3	53.05	53.64	53.69	53.80	53.82	52.52	53.15	53.19	53.32	53.33
F2, M3, T4	53.09	53.68	53.72	53.84	53.86	52.60	53.23	53.27	53.39	53.41

Where:
F1 is the logarithmic and F2 – geometric cooling scheme; M1 stands for Swap, M2 – for Adjacent Swap and M3 – for Insert; initial temperatures: T1 = 5, T2 = 1, T3 = 0.5, and T4 = 0.1; resource allocation algorithms: A1 – no shift (only first stage of proposed algorithm is performed), A2 – BasicChaining, A3 – ISH, A4 – ISH^2, and A5 – ISH^2-UA

Table 3. *Average values of objective function F for various parameters of the algorithm and for 90-activity projects (set J90)*

Algorithm parameters	Serial SGS					Parallel SGS				
	A1	A2	A3	A4	A5	A1	A2	A3	A4	A5
F1, M1, T1	31.33	31.85	31.95	32.19	32.23	31.39	31.93	32.04	32.28	32.32
F1, M1, T2	31.27	31.80	31.90	32.14	32.18	31.33	31.87	31.98	32.22	32.26
F1, M1, T3	31.36	31.89	31.99	32.22	32.26	31.40	31.94	32.05	32.29	32.33
F1, M1, T4	31.28	31.81	31.91	32.14	32.19	31.36	31.90	32.01	32.25	32.29
F1, M2, T1	30.27	30.83	30.95	31.20	31.23	30.92	31.48	31.59	31.84	31.88
F1, M2, T2	30.28	30.83	30.94	31.18	31.23	30.83	31.39	31.50	31.75	31.79
F1, M2, T3	30.43	30.98	31.09	31.33	31.38	30.89	31.45	31.56	31.81	31.85
F1, M2, T4	30.30	30.88	30.99	31.23	31.27	30.84	31.40	31.51	31.75	31.80
F1, M3, T1	31.94	32.45	32.56	32.79	32.82	31.57	32.11	32.22	32.45	32.49
F1, M3, T2	31.81	32.33	32.44	32.67	32.71	31.66	32.20	32.31	32.54	32.58
F1, M3, T3	31.85	32.36	32.47	32.70	32.73	31.58	32.12	32.22	32.46	32.50
F1, M3, T4	31.86	32.38	32.48	32.71	32.74	31.62	32.15	32.26	32.49	32.53
F2, M1, T1	30.44	30.99	31.11	31.36	31.41	31.01	31.56	31.67	31.91	31.96
F2, M1, T2	31.08	31.63	31.74	31.98	32.02	31.28	31.83	31.94	32.19	32.23
F2, M1, T3	31.28	31.82	31.94	32.18	32.22	31.36	31.91	32.02	32.26	32.30
F2, M1, T4	31.34	31.88	31.99	32.22	32.26	31.38	31.93	32.04	32.28	32.32
F2, M2, T1	29.93	30.50	30.62	30.87	30.92	30.66	31.21	31.33	31.58	31.62
F2, M2, T2	30.04	30.60	30.72	30.96	31.00	30.77	31.33	31.44	31.69	31.73
F2, M2, T3	30.23	30.79	30.91	31.16	31.20	30.87	31.43	31.54	31.79	31.84
F2, M2, T4	30.34	30.90	31.01	31.26	31.30	30.86	31.41	31.53	31.78	31.82
F2, M3, T1	30.58	31.15	31.26	31.51	31.55	31.15	31.71	31.82	32.07	32.12
F2, M3, T2	31.44	31.99	32.10	32.34	32.38	31.44	31.99	32.10	32.33	32.38
F2, M3, T3	31.66	32.20	32.31	32.55	32.59	31.56	32.11	32.22	32.46	32.50
F2, M3, T4	**31.96**	**32.47**	**32.58**	**32.81**	**32.85**	31.70	32.24	32.35	32.58	32.62

Symbols as in Table 2

6. SUMMARY

This paper deals with the problem of maximisation of discounted cash flows for a project settled by milestones. A solution improvement algorithm is presented, using iterative unit right shifts of activities, designed to optimise the objective function. Shifts are performed for an assumed resource allocation to individual activities.

Extensive experiments and comparison of our algorithm with other approaches will be per-formed in further authors' works. The results of the presented preliminary computational experiments confirm the effectiveness of the two-phase algorithm proposed. They show the selection of appropriate resource allocation to be particularly important. Accordingly, in their further research into this area, the authors intend to, *inter alia*, develop an algorithm of local search for the resource allocation problem, with a view to identify optimum right shifts of the activities.

The problem considered here is of current interest and essential importance from the perspective of practical applications. The model proposed by us can be used in the execution of large-scale production, construction, or development projects.

REFERENCES

Baroum S.M., Patterson J.H., 1996. The development of cash flow weight procedures for maximizing the net present value of a project. *Journal of Operations Management*, **14**(3), pp. 209–227.

Błażewicz J., Lenstra J., Kan A.R., 1983. Scheduling subject to resource constraints – classification and complexity. *Discrete Applied Mathematics*, **5**, pp. 11–24.

Boctor F.F., 1996. Resource-constrained project scheduling by simulated annealing. *International Journal of Operational Research*, **34**(8), pp. 2335–2351.

Bouleimen K., Lecocq H., 2003. A new efficient simulated annealing algorithm for the resource constrained project scheduling problem and its multiple version. *European Journal of Operational Research*, **149**, pp. 268–281.

Deblaere F., Demeulemeester E.L., Herroelen W.S., Van De Vonder S., 2006. *Proactive resource allocation heuristics for robust project scheduling*. Working Paper KBI_0608, Leuven.

Hartmann S., Briskorn D., 2012. A Survey of Variants and Extensions of the Resource-Constrained Project Scheduling Problem. *European Journal of Operational Research*, **207**(1), pp. 1–14.

Hartmann S., Kolisch R., 2000. Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem. *European Journal of Operational Research*, **127**, pp. 394–407.

He Z., Xu Y., 2008. Multi-mode project payment scheduling problems with bonus penalty structure. *European Journal of Operational Research*, **189**, pp. 1191–1207.

He Z., Wang N., Jia T., Xu Y., 2009. Simulated annealing and tabu search for multi-mode project payment scheduling. *European Journal of Operational Research*, **198**(3), pp. 688–696.

He Z., Wang N., Jia T., Xu Y., 2012. Metaheuristics for multi-mode capital-constrained project payment scheduling. *European Journal of Operational Research*, **223**, pp. 605–613.

Herroelen W., Reyck B. D., Demeulemeester E., 1997. Project network models with discounted cash flows: A guided tour through recent developments. *European Journal of Operational Research*, **100**, pp. 97–121.

Icmeli O., Erenguc S., 1996. The resource constrained time/cost tradeoff project scheduling problem with discounted cash flows. *Journal of Operation Management*, **14**, pp. 255–275.

Kimms A., 2001. Maximizing the net present value of a project using a Lagrangian relaxation based heuristic with tight upper bounds. *Annals of Operations Research*, **102**, pp. 221–236.

Kirkpatrick S., Gelatt C.D., Vecchi M.P., 1983. Optimization by simulated annealing. *Science*, **220**, pp. 671–680.

Klimek M., 2010. *Predyktywno-reaktywne harmonogramowanie produkcji z ograniczoną dostępnością zasobów* (Predictive-Reactive Scheduling of Production with Limited Availability of resources). Ph.D. Dissertation, AGH, Kraków, Poland.

Klimek M., Łebkowski P., 2010. Resource allocation for robust project scheduling. *Bulletin of the Polish Academy of Sciences Technical Sciences*, **59**(1), pp. 51–55.

Kolisch R., 1996. Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. *European Journal of Operational Research*, **90**, pp. 320–333.

Kolisch R., Padman R., 2001. An integrated survey of deterministic project scheduling. *OMEGA The International Journal of Management Science*, **29**, pp. 249–272.

Kolisch R., Sprecher A., 1997. PSPLIB – a project scheduling library. *European Journal of Operational Research*, 96, pp. 205–216.

Leus R., 2003. *The generation of stable project plans.* PhD Dissertation, K.U. Leuven.

Mika M., Waligóra G., Węglarz J., 2005. Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models. *European Journal of Operational Research*, **164**(3), pp. 639–668.

Pinder J.P., Marucheck A.S., 1996. Using discounted cash flow heuristics to improve project net present value. *Journal of Operations Management*, **14**, pp. 229–240.

Policella N., 2005. *Scheduling with Uncertainty – A Proactive Approach using Partial Order Schedules.* Ph.D. Dissertation, La Sapienza Universita, Rome.

Policella N., Oddi A., Smith S., Cesta A., 2004. Generating robust partial order schedules, In *Proceedings of CP2004*, Toronto.

Russell A.H., 1970. Cash flows in networks. *Management Science*, **16**, pp. 357–373.

Selle T., Zimmermann J., 2003. A bidirectional heuristic for maximizing the net present value of large-scale projects subject to limited resources. *Naval Research Logistics*, **50**, pp. 130–148.

Ulusoy G., Özdamar L., 1995. A heuristic scheduling algorithm for improving the duration and net present value of a project. *International Journal of Operations and Production Management*, **15**, pp. 89–98.

Ulusoy G., Sivrikaya-Serifoglu F., Sahin S., 2001. Four Payment Models for the Multi-Mode Resource Constrained Project Scheduling Problem with Discounted Cash Flows, *Annals of Operations Research*, **102**, pp. 237–261.

Vanhoucke M., 2006. *A scatter search procedure for maximizing the net present value of a resource-constrained project with fixed activity cash flows.* Working Paper 2006/417, Gent, pp. 1–23.

Vanhoucke M., Demeulemeester E., Herroelen W., 2001. Maximizing the net present value of a project with linear time-dependent cash flows. *International Journal of Production Research*, **39**(14), pp. 3159–3181.

Węglarz J. (editor), 1999. *Project Scheduling: Recent Models, Algorithms and Applications.* Kluwer Academic Publishers.