A linear-discrete scheduling model for the resource-constrained project scheduling problem

Pierre Bonnal, Didier Gourc, Ari-Pekka Hameri, Germain Lacoste

A linear-discrete scheduling model for the resource-constrained project scheduling problem

PIERRE BONNAL[1]*, DIDIER GOURC[2], ARI-PEKKA HAMERI[3] and GERMAIN LACOSTE[4]

[1]*CERN, CH1211 Geneva 23, Switzerland*
[2]*ENSTIMAC, F81013 Albi, France*
[3]*UNIL-HEC, CH1015 Lausanne, Switzerland*
[4]*ENIT, F65016 Tarbes, France*

Received 16 July 2003; accepted 26 November 2004

For some specific types of construction projects, the classical CPM or PDM scheduling techniques are not the most suitable. Few specific scheduling approaches have been developed to cope with construction projects that are made of either repetitive activities or activities with linear developments. But real-world construction projects do not consist only of such activities. They are generally made of a mixture of linear and/or repetitive activities and of more conventional activities. To allow this, the linear scheduling problem is reformulated, so classical schedule calculation approaches can be used. The implementation of some Allen's algebra features to avoid adverse discontinuities and to allow crew/work continuity, together with a resource-driven and space-constrained scheduling are among the key features of the proposed approach. It is also a spin-off of off-the-field practices used for scheduling real projects in the particle accelerator construction domain; an excerpt from such a construction project is provided for illustrating the methodology.

Keywords: Construction scheduling, linear scheduling, optimization

Introduction

Over the last two decades, several articles (Selinger, 1980; Johnston, 1981; Chrzanowski and Johnston, 1986; Handa and Barcia, 1986; Russell and Caselton, 1988; Eldin and Senouci, 1994, 2000; Harmelink and Rowings, 1998; Harmelink, 2001; Moselhi and Hassanein, 2003) have been published to address the scheduling problem of construction or installation projects which typically have a linear development. Most of these papers refer to the so-called Linear Scheduling Model (LSM) that is an alternate methodology to the 40-year-old Critical Path Method (CPM). The linear scheduling methods that are presented in these papers do not allow considering systematically resources except the spatial one. In the present article, we have reformulated the problem so it can be treated as a resource-constrained project scheduling problem. Repetitive construction scheduling problems are a class of problems similar to those that can benefit from the LSM approach; the approach presented in the present article can also be seen as a generalization of the repetitive construction scheduling to the linear scheduling.

The projects concerned by the LSM methodology are projects that consist of a majority of linear activities. Harmelink and Rowings (1998) give the following definition to the latter: linear activities are those activities that are completed as they progress along a path. For instance, the digging of a railway tunnel, the repaving of a motorway or the installation and interconnection works of a particle accelerator, are typically projects that belong to the family of linear development projects.

The implementations of LSM-like approaches on real life projects have demonstrated their efficiency. From a practitioner point of view, the benefits of using the approach stem from:

- a smaller breakdown of activities is sufficient to perform the activity network analysis; and
- the schedules issued are more concise, which improve communication.

Among the scheduling approaches suited to a particular class of construction project, one should also mention

*Author for correspondence. E-mail: pierre.bonnal@cern.ch

all the approaches proposed for dealing with repetitive projects (Ashley, 1980; Kavanagh, 1985; O'Brien *et al.*, 1985; Reda, 1990; Russell and Wong, 1993; El-Rayes and Moselhi, 2001; El-Rayes, 2001; El-Rayes *et al.*, 2002) and the Line-of-Balance approach, proposed by Fouch in the 1940s to respond to industrial scheduling problems at Goodyear Tire & Rubber Co. (see Suhail and Neale; 1994, for instance). The LSM approaches differ from the latter by the fact that they are applicable to activities with a pure linear development that cannot be systematically discretized. For instance, the notion of start and finish stations is something that is specific to LSM.

In terms of development, repetitive scheduling approaches have certainly reached a higher level of maturity. Several successful attempts have been made to make schedule analyses systematic: CPM-like calculation with forward and backward passes, resource-driven scheduling (Moselhi and El-Rayes, 1993; Russell and Wong, 1993; Suhail and Neale, 1994; El-Rayes and Moselhi, 1998; El-Rayes, 2001; Leu and Hwang, 2001; El-Rayes *et al.*, 2002). LSM approaches have not yet benefited of such functionalities; LSM schedule calculations are often limited to graphical analyses.

As mentioned by El-Rayes (2001), scheduling of repetitive construction projects can be significantly improved by considering three main practical requirements. These requirements are (1) the application of resource-driven scheduling that enable crew/work continuity, (2) the minimization of the project duration by an optimized utilization of the resources available, and (3) the integration in a single activity network of activities on different types: repetitive and non-repetitive activities for instance.

The aim of the present article is to describe a scheduling methodology that is suitable to projects that have activities with linear developments, and that integrates the three practical requirements mentioned here before. To do so, the problem has been formulated in such a way resource-constrained scheduling methodologies can be implemented. Several types of activities are supported: activities that are space-constrained can be mixed with activities that are not; activities with a linear development can be mixed with discrete activities. Some Allen's algebra features (Allen, 1983) are implemented for addressing the crew/work continuity issue that is also mentioned here before.

The scheduling methodology presented in this article is henceforth called LDSM (Linear-Discrete Scheduling Model) for distinguishing it among the many scheduling techniques that are available for addressing this type of construction problem.

The rest of the article is organized in the following way: definitions are given in section 2. Then an approach for solving a linear-discrete scheduling (LDS) problem is proposed in section 3. Section 4 details the relevant algorithms. Before conclusion, this approach is applied to a real case problem (section 5).

Definitions and introductory remarks

Space-constrained vs. non-space-constrained activities

Space-constrained activities can be defined as activities that rely on the availability of a scarce spatial resource for being schedules, and then executed. Non-space-constrained activities are activities for which spatial resources do not need to be considered to have them scheduled, and then executed. In the framework of such construction project, space-constrained activities are in general the construction/installation activities carried out on the construction site (excavation works, digging works, pipe works, cable pulling, etc.) while non-space-constrained activities are those performed offsite (engineering works, etc.) or at supplier or contractor premises (prefabrication, etc.).

Linear vs. discrete-space-constrained activities

Linear space-constrained activities have been defined here before: these are activities that progress along a path; they generally use this 'path' a renewable resource, e.g. the digging of a railway tunnel, the paving of a motorway, etc. Discrete-space-constrained activities are activities that also use a spatial renewable resource, but that do not progress along a path, e.g. the digging of a small alcove in a railway tunnel, the earthwork for a junction at the crossing of two motorways, etc.

Linear-discrete schedule diagramming

Above all, schedules are communication media even if the underlying optimization mechanisms are of prime importance. Gantt charts are known to be the best means for communicating the temporal information of a project made of non-space-constrained activities. On these charts, the time goes along the horizontal axis from left to right, while the Work Breakdown Structure (WBS) depicting in a hierarchical way all the activities to perform to complete the project is viewed vertically. Authors of LSM have preferred the one-dimensional space resource viewed over the horizontal axis, while the time runs along the vertical one.

For convenience, and especially for facilitating the communication of LDSM schedules among people certainly more familiar with Gantt charts, the Gantt's scheme is preferred: the time running along the

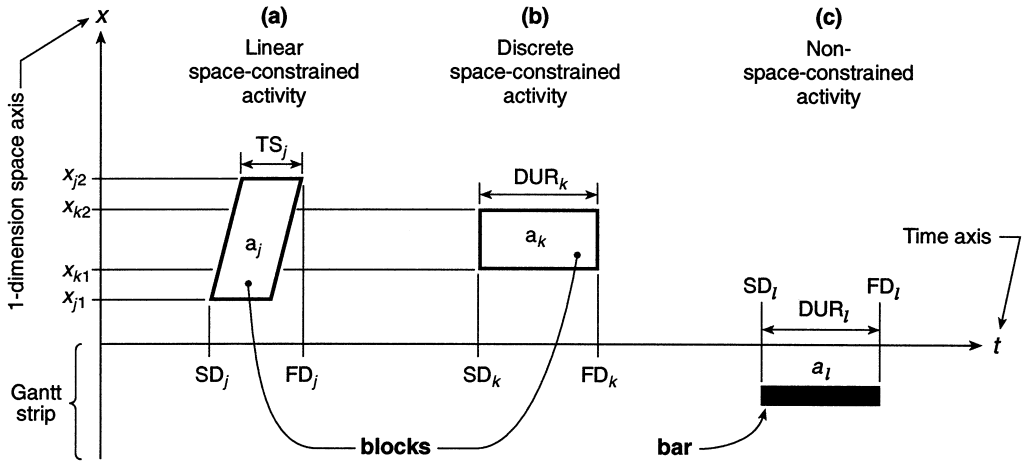

Figure 1 The three types of activities LDSM incl. naming conventions

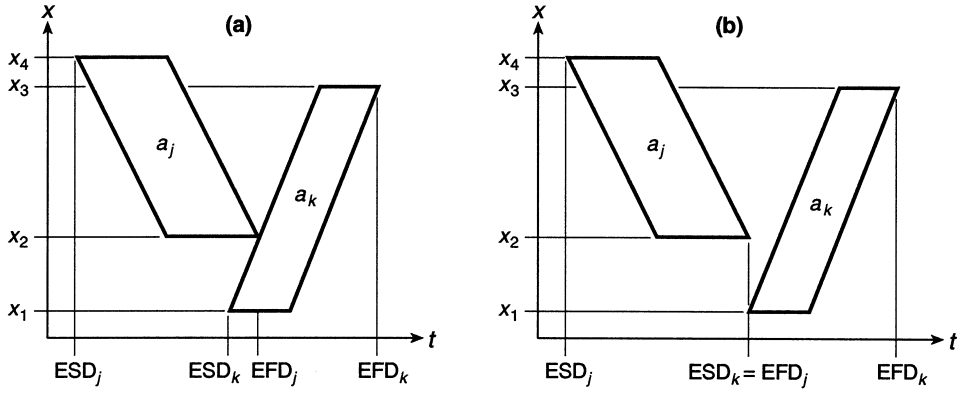

Figure 2 Optimally calculated vs. CPM calculated LDS diagram – example no. 1

horizontal axis and the one-dimensional space stations spread along the vertical axis.

For compactness and for complying with LSM principles, also shared by the critical chain approach (Newbold, 1998), several non-overlapping activities can be merged on a same row. Both space-constrained activities and non-space-constrained activities are segregated into distinct horizontal strips.

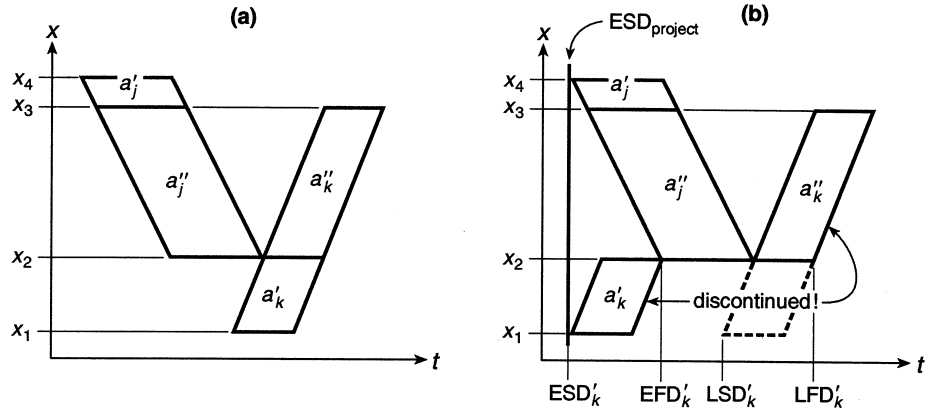

Figure 3 Fragmented LDS diagram; calculated fragmented LDS diagram

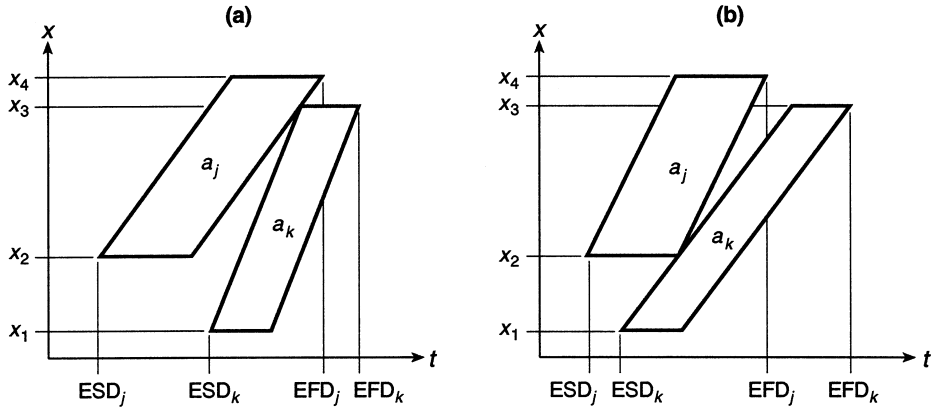

Figure 4 CPM calculated LDS diagram – example no. 2

The typology of LDSM activities

Basically, LDSM supports three types of activities:

- linear space-constrained activities, resource-constrained or not;
- discrete space-constrained activities, resource-constrained or not; and
- and other activities, discrete, non-space-constrained, resource-constrained or not.

In the LDSM vocabulary, all space-constrained activities are called *blocks*, while others are called *bars*, in reference to bar charts, a name used for Gantt charts (cf. Figure 1). Blocks can be of two types: parallelogram-shaped blocks referring to linear activities and

Symbol and inverse	Meaning	Relations between dates	Gantt chart representation
$a_j < a_k$ $a_k > a_j$	"before" "after"	$SD_j < FD_j < SD_k < FD_k$	
a_j **m** a_k a_k **mi** a_j	"meets"	$SD_j < FD_j = SD_k < FD_k$	
a_j **o** a_k a_k **oi** a_j	"overlaps"	$SD_j < SD_k < FD_j < FD_k$	
a_j **s** a_k a_k **si** a_j	"starts"	$SD_j = SD_k < FD_j < FD_k$ "by convention"	
a_j **d** a_k a_k **di** a_j	"during"	$SD_k < SD_j < FD_j < FD_k$	
a_j **f** a_k a_k **fi** a_j	"finishes"	$SD_k < SD_j < FD_j = FD_k$ "by convention"	
$a_j = a_k$	"equals"	$SD_j = SD_k < FD_j = FD_k$	

Figure 5 Allen's temporal logic relations

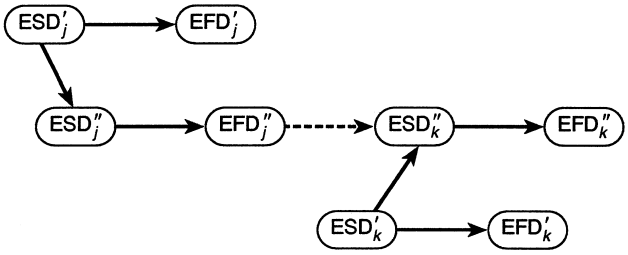

Figure 6 Forward pass DG of example no. 1

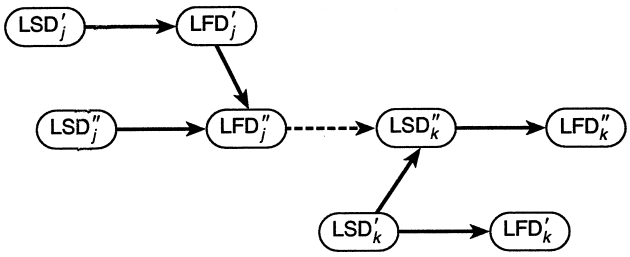

Figure 7 Backward pass DG of example no. 1

rectangle-shaped blocks that correspond to discrete activities.

Linear space-constrained activities (cf. Figure 1a)

Let a_j be a linear space-constrained activity, in addition to precedence and resource constraints, such an activity is characterized by a 6-tuple (x_{j1}, x_{j2}, θ_j, TS_j, Γ_j^{-1}, $\mathbf{r}_j$) with:

- x_{j1}: start station of activity a_j
- x_{j2}: finish station of activity a_j
- θ_j: production rate of activity a_j
- TS_j: temporal span associated with activity a_j
- Γ_j^{-1}: set of predecessors of activity a_j ; and $\Gamma_j^{-1}=\emptyset$ if a_j has no predecessor
- $\mathbf{r}_j$: vector of the resources required for completing activity a_j.

The production rate θ_j is the spatial progress foreseen per unit of time: typically, production rates can be expressed in feet/hour, km/week, units/day… θj is positive if a_j progresses along x ascending, i.e. if $x_{j1}<x_{j2}$; otherwise θ_j is negative. The true production rate capacity is based on $|\theta_j|$ and not on θ_j.

If the start date SD_j of activity a_j is known, the finish date FD_j of this activity can be obtained as follows:

$$FD_j = SD_j + (x_{j2} - x_{j1})/\theta_j + TS_j. \quad (1)$$

It is also important to remark that the definition to predecessor activity in the LDS context differs to the one of predecessors in a CPM context. To illustrate this, let a_j and a_k be two discrete activities, such as $\Gamma_j^{-1}=\{a_j\}$. In a CPM context, this means that a_k cannot start until a_j has ended. In the LDS environment, precedence constraints generally means that an activity can start, as soon as its predecessor activities has ended in the vicinity of its start station.

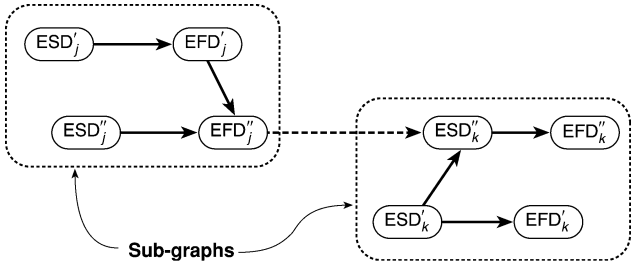

Figure 8 Forward date graph with sub-graphs highlighted

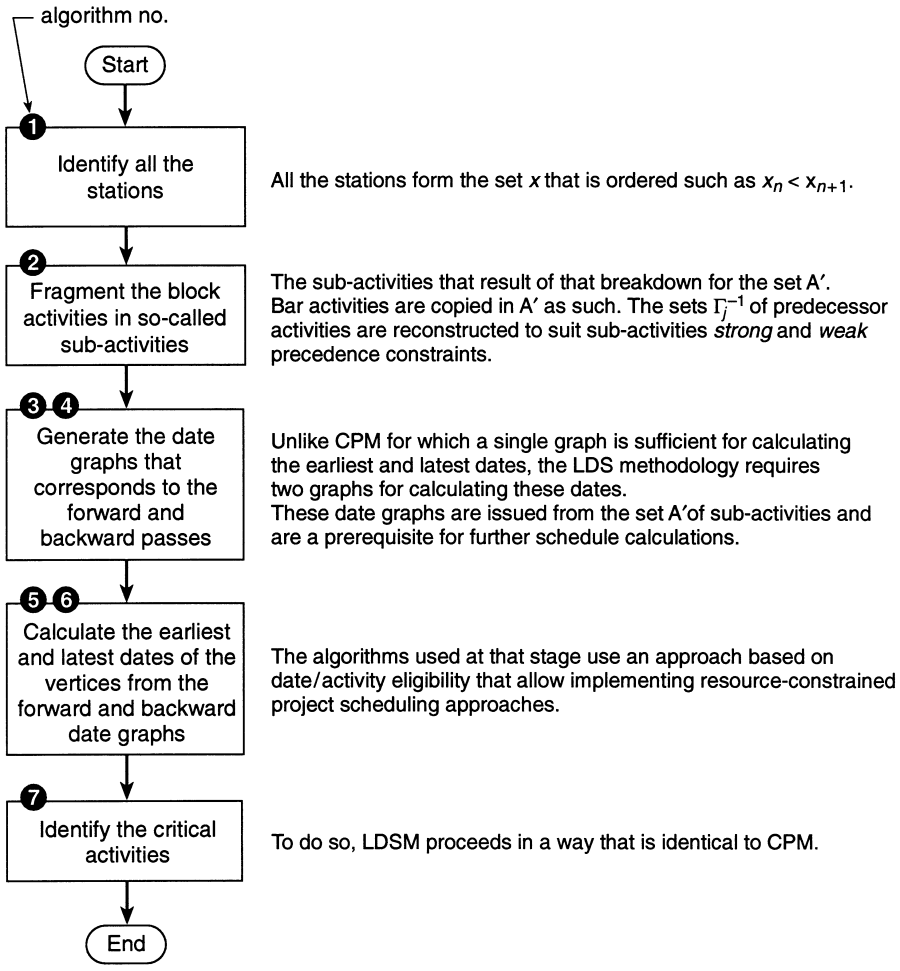

Figure 9 Overview of the LDSM computational procedure

Discrete space-constrained activities (cf. Figure 1b)

Such an activity is also characterized by a 6-tuple (x_{j1}, x_{j2}, ∞, DUR_j, Γ_j^{-1}, $\mathbf{r}_j$) with:

x_{j1}: start station of activity a_j
x_{j2}: finish station of activity a_j
DUR_j: duration of activity a_j: $DUR_j = FD_j - SD_j$
Γ_j^{-1}: set of predecessors of activity a_j ; and $\Gamma_j^{-1} = \emptyset$ if a_j has no predecessor
$\mathbf{r}_j$: vector of the resources required for completing activity a_j.

A discrete space-constrained activity is a linear space-constrained activity to which $\theta_j = \infty$ and $TS_j = DUR_j$. In the remaining of this article, we won't make any distinction anymore between linear and discrete space-constrained activities. They will be simply called block activities.

Non-space-constrained activities (cf. Figure 1c)

In addition to precedence and resource constraints, the duration is sufficient to describe a non-space-constrained activity, henceforth called as bar activity.

Calculated dates

The aim of a critical path analysis is to determine for all the activities of a network the earliest and latest start and finish dates: ESD_j, EFD_j, LSD_j and LFD_j. The critical activities are those for which earliest and latest dates are identical, i.e. $ESD_j = LSD_j$ or $EFD_j = LFD_j$. CPM, LSM and LDSM do share the same objective.

An approach for solving a LDS problem

Several approaches can be found to solve a LDS problem. We take two examples to find out where the solving difficulties are located.

Activity number and label	Linear	Discrete	Bar act.	Stations x_{j1} [m]	Stations x_{j2} [m]	rate / span / duration θ_{j2} [m/week]	rate / span / duration TS_j/DUR_j [weeks]	Predecessors Γ_j^{-1} (act., type, LAG_j)
01 Mark position of equipment and services		•		20 265	23 270	∞	12	2001/10/15
02 Repair floor and drill holes between UA83 and RA83		•		20 265	23 270	∞	10	(**01**, **FS**, 0)
03 Re-assemble metallic structure of RR77			•	—	—	—	4	2002/04/08
04 Install metallic structure in RR77		•		20 265	20 275	∞	2	(**02**, **FS**, 0) (**03**, **FS**, 0)
05 Procure electrical equipment for alcoves			•	—	—	—	8	2002/02/18
06 Refurbish electrical alcove RE78		•		20 935	20 940	∞	4	(**02**, **FS**, 0) (**05**, **FS**, 0)
07 Refurbish electrical alcove RE82		•		22 380	22 385	∞	4	(**02**, **FS**, 0) (**06**, **FS**, 0)
08 Install cable trays and pull AC cables in R78 and R79	•			20 265	21 660	200	2	(**04**, **FS**, 0) (**06**, **FS**, 0)
09 Install cable trays and pull AC cables in R81 and R82	•			21 660	23 270	200	2	(**07**, **FS**, 0)
10 Install and weld new water, N_2 and He pipes	•			20 265	23 270	250	3	(**02**, **FS**, 0) (**09**, **FS**, 0)
11 Pull DC cables in R78 and R79	•			20 265	21 660	175	2	(**10**, **FS**, 0)
12 Pull DC cables in R81 and R82	•			21 660	23 045	175	2	(**10**, **FS**, 0)
13 Install electrical sub-station in UA83		•		23 045	23 270	∞	3	(**12**, **FS**, 0)
14 Pull water cooled cables between UA83 and RA83		•		23 045	23 270	∞	2	(**13**, **FS**, 0)
15 Pull signal cables in R78 and R79	•			20 265	21 660	100	4	(**11**, **FS**, 0)
16 Pull signal cables in R81 and R82	•			21 660	23 270	100	4	(**14**, **FS**, 0)

Note: "YYYY/MM/DD" means: activity cannot start before YYYY/MM/DD

Figure 10 Example-description of the activities

Activity no.	Sub-activity	Stations x_{j1} [m]	Stations x_{j2} [m]	Predecessors Γ_j^{-1} (act., type, LAG_j)
01	01.1	20 265	20 275	2001/10/15
	01.2	20 275	20 935	(01.1,=,0)
	01.3	20 935	20 940	(01.2,=,0)
	01.4	20 940	21 660	(01.3,=,0)
	01.5	21 660	22 380	(01.4,=,0)
	01.6	22 380	22 385	(01.5,=,0)
	01.7	22 385	23 045	(01.6,=,0)
	01.8	23 045	23 270	(01.7,=,0)
02	02.1	20 265	20 275	(01.1,**FS**,0)
	02.2	20 275	20 935	(02.1,=,0)(01.2,**FS**,0)
	02.3	20 935	20 940	(02.2,=,0)(01.3,**FS**,0)
	02.4	20 940	21 660	(02.3,=,0)(01.4,**FS**,0)
	02.5	21 660	22 380	(02.4,=,0)(01.5,**FS**,0)
	02.6	22 380	22 385	(02.5,=,0)(01.6,FS,0)
	02.7	22 385	23 045	(02.6,=,0)(01.7,**FS**,0)
	02.8	23 045	23 270	(02.7,=,0)(01.8,**FS**,0)
03	03.1	—	—	2002/04/08
04	04.1	20 265	20 275	(02.1,**FS**,0)(03.1,**FS**,0)
05	05.1	—	—	2002/02/18
06	06.1	20 935	20 940	(02.3,**FS**,0)(05.1,**FS**,0)
07	07.1	22 380	22 385	(02.3,**FS**,0)(05.1,**FS**,0)
08	08.1	20 265	20 275	(04.1,**FS**,0)
	08.2	20 275	20 935	(08.1,**O**,...)
	08.3	20 935	20 940	(08.2,**O**,...)(06.1,**FS**,0)
	08.4	20 940	21 660	(08.3,**O**,...)
09	09.1	21 660	22 380	—
	09.2	22 380	22 385	(09.1,**O**,...)(07,**FS**,0)
	09.3	22 385	23 045	(09.2,**O**,...)
	09.4	23 045	23 270	(09.3,**O**,...)
10	10.1	20 265	20 275	(08.1,**FF**,3)
	10.2	20 275	20 935	(10.1,**O**,...)(08.2,**FF**,3)
	10.3	20 935	20 940	(10.2,**O**,...)(08.3,**FF**,3)
	10.4	20 940	21 660	(10.3,**O**,...)(08.4,**FF**,3)
	10.5	21 660	22 380	(10.4,**O**,...)(09.1,**FF**,3)
	10.6	22 380	22 385	(10.5,**O**,...)(09.2,**FF**,3)
	10.7	22 385	23 045	(10.6,**O**,...)(09.3,**FF**,3)
	10.8	23 045	23 270	(10.7,**O**,...)(09.4,**FF**,3)
11	11.1	20 265	20 275	(10.1,**SS**,3)
	11.2	20 275	20 935	(11.1,**O**,...)(10.1,**SS**,3)
	11.3	20 935	20 940	(11.2,**O**,...)(10.3,**SS**,3)
	11.4	20 940	21 660	(11.3,**O**,...)(10.4,**SS**,3)
12	12.1	21 660	22 380	(10.5,**SS**,3)
	12.2	22 380	22 385	(11.1,**O**,...)(10.6,**SS**,3)
	12.3	22 385	23 045	(11.2,**O**,...)(10.7,**SS**,3)
13	13.1	23 045	23 270	(12.3,**FS**,0)(10.8,**FS**,0)
14	14.1	23 045	23 270	(13.1,**FS**,0)
15	15.1	20 265	20 275	(11.1,**SS**,2)
	15.2	20 275	20 935	(15.1,**O**,...)(11.2,**SS**,2)
	15.3	20 935	20 940	(15.2,**O**,...)(11.3,**SS**,2)
	15.4	20 940	21 660	(15.3,**O**,...)(11.4,**SS**,2)
16	16.1	21 660	22 380	(12.1,**SS**,2)
	16.2	22 380	22 385	(16.1,**O**,...)(12.2,**SS**,2)
	16.3	22 385	23 045	(16.2,**O**,...)(12.3,**SS**,2)
	16.4	23 045	23 270	(16.3,**O**,...)(14.1,**FS**,0)

Note: "YYYY/MM/DD" means: activity cannot start before YYYY/MM/DD

Figure 11 Example-breakdown of the activities into sub-activities

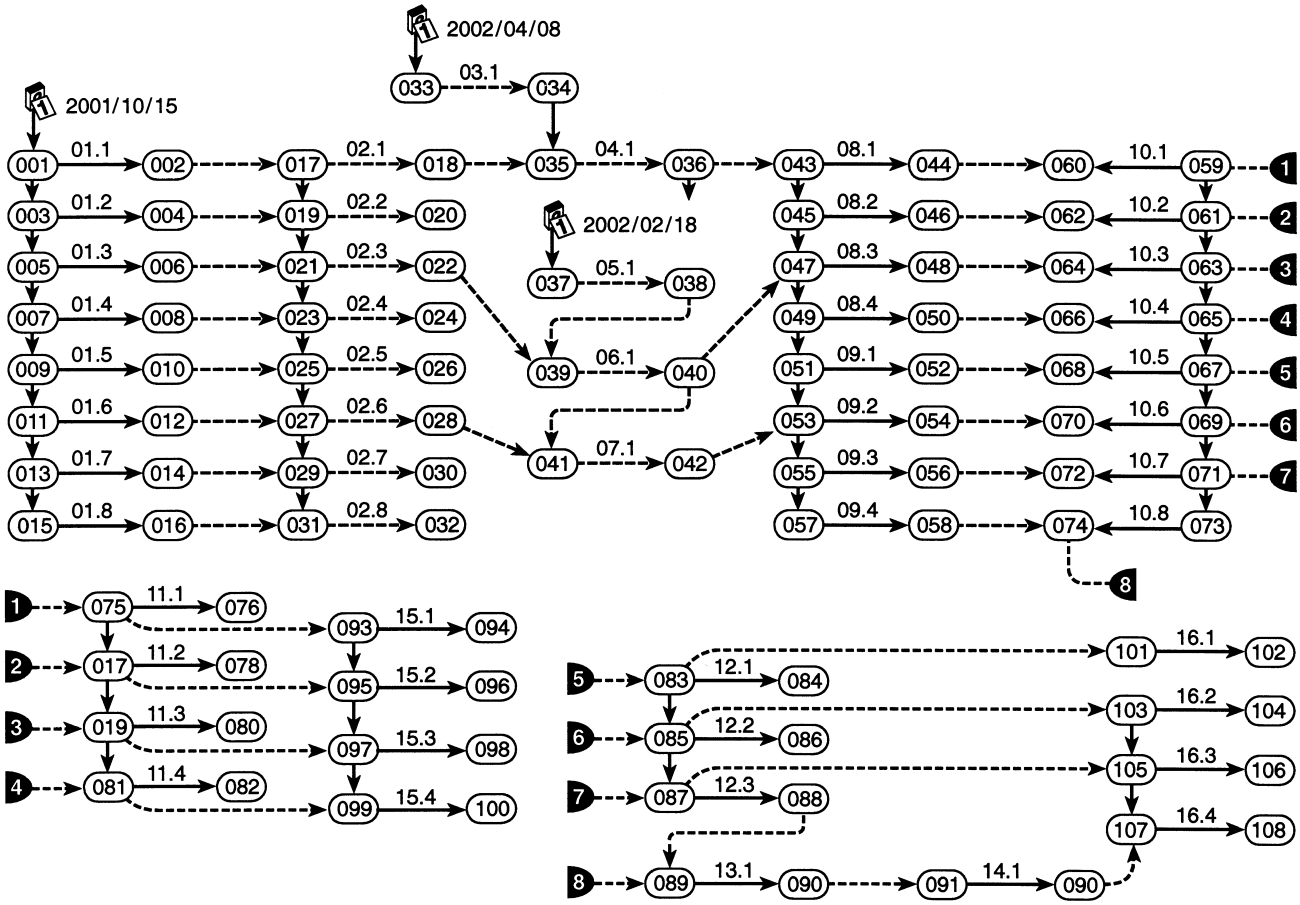

Figure 12 Example-forward pass DG

Example no. 1

Let aj and ak be two block activities such as $a_j=(x_4,x_2,\theta_j,TS_j,\varnothing,\varnothing)$ and $a_k=(x_1,x_3,\theta_k,TS_k,\{a_j\},\varnothing)$, with $x_1<x_2<x_3<x_4$, i.e. $\theta_j<0$ and $\theta_k>0$. Figure 2a shows how these two activities should be scheduled optimally, i.e. as soon as possible. It can be observed that the early finish date EFD_j of activity a_j is scheduled later than the early start date ESD_k of activity a_k. Figure 2b shows how CPM would have scheduled these two activities.

The approach we are proposing in this article for solving such a problem requires a systematic fragmentation of the space-constrained activities along the space axis. This means for instance that activities aj and ak would be broken down into four sub-activities a_j', a_j'', a_k' and a_k'' such as: $a_j'=(x_4, x_3, \theta_j, TS_j, \varnothing, \varnothing)$,

$a_j''=(x_3, x_2, \theta_j, TS_j, \{aj'\}\ \varnothing)$, $a_k'=(x_1, x_2, \theta_k, TS_k, \varnothing, \varnothing)$, $a_k''=(x_2, x_3, \theta_k, TS_k, \{a_k', a_j''\}, \varnothing)$ (see Figure 3a).

The precedence constraints of this set of sub-activities are the following: $a_j' \prec a_j''$, $a_k' \prec a_k''$ and $a_j'' \prec a_k''$ (the 'x $\prec$ y' stands for 'x precedes y'). The results of a CPM calculation are given in Figure 3b.

In some situations, such schedule can be acceptable. But in most real-world situations, project management practitioners would prefer a continuity in the performing of activity a_k, i.e. baselining LSD_k' and LFD_k' instead of ESD_k' and EFD_k'. Implementation of features of Allen's temporal logic relation leads to a calculation procedure that fulfills real-world requirements, especially the possibility of mixing earliest and latest dates when needed, and by the way responding to the crew/work continuity requirement.

Example no. 2

Let a_j and a_k be two activities such as $a_j=(x_2, x_4, \theta_j, TS_j, \varnothing, \varnothing)$ and $a_k=(x_1, x_3, \theta_k, TS_k, \{aj\}, \varnothing)$, with $x_1<x_2<x_3<x_4$, i.e. $\theta_j>0$ and $\theta_k>0$.

Three cases shall be distinguished: $\theta_j<\theta_k$ (cf. Figure 4a), $\theta_j>\theta_k$ (cf. Figure 4b) or $\theta_j=\theta_k$. In all cases, a_k must be scheduled so it just does not interfere with

a_j. After fragmentation, the sub-activities sharing the same space-strip are scheduled using the following formulae:

if $\theta_j \leqslant \theta_k$:

$$SD_k'' = SD_j' + TS_j + (x_3 - x_2)/\theta_j - (x_3 - x_2)/\theta_k \quad (2)$$

$$FD_k'' = FD_j' + TS_k \quad (3)$$

if $\theta_j \geqslant \theta_k$:

$$SD_k'' = SD_j' + TS_j \quad (4)$$

$$FD_k'' = FD_j' + TS_k + (x_3 - x_2)/\theta_k - (x_3 - x_2)/\theta_j \quad (5)$$

Extended set of precedence constraints

In his temporal algebra, Allen (1983) has proposed eight relations (and 13 if their inverses are considered) for describing all the possible configurations of two temporal intervals. These 13 relations are presented in Figure 5.

The four types of precedence constraints of the precedence method can be defined using Allen's formalism as follows (let a_j and a_k be two activities; symbol $\vee$ stands for 'or'):

$$a_j \underset{\text{finish-start}}{\prec} a_k \equiv a_j(\mathbf{m} \vee <) a_k$$

$$a_j \underset{\text{start-start}}{\prec} a_k \equiv a_j(\mathbf{si} \vee = \vee \mathbf{s} \vee \mathbf{di} \vee \mathbf{fi} \vee \mathbf{o} \vee \mathbf{m} \vee <) a_k$$

$$a_j \underset{\text{start-finish}}{\prec} a_k \equiv a_j(\mathbf{mi} \vee \mathbf{oi} \vee \mathbf{f} \vee \mathbf{d} \vee \mathbf{si} \vee = \vee \mathbf{s} \vee \mathbf{di} \vee \mathbf{fi} \vee \mathbf{o} \vee \mathbf{m} \vee <) a_k$$

$$a_j \underset{\text{finish-finish}}{\prec} a_k \equiv a_j(\mathbf{f} \vee = \vee \mathbf{fi} \vee \mathbf{d} \vee \mathbf{s} \vee \mathbf{o} \vee \mathbf{m} \vee <) a_k$$

Let a_j and a_k be two space-constrained activities, the Allen's temporal logic relations that best describe the precedence constraints featured in a fragmented LDS diagram are:

$$a_j(\mathbf{o} \vee \mathbf{m} \vee <) a_k \equiv \text{finish-start precedence constraint}$$

$$a_k'(\mathbf{o} \vee =) a_k'' \equiv \text{precedence constraint in a fragmented activity}$$

Equivalent date graph

The use of a date graph (DG) – or point graph (Zaidi, 2001) – is convenient to solve an activity network that involves Allen's temporal logic relations. A date diagram is a valued digraph in which nodes (vertices) feature the dates to calculate – also called time stamps in some textbooks and articles – and arcs (edges) the time intervals that separate dates that are directly dependent.

In the framework of the LDS problem, two types of arcs are distinguished:

- strong intervals, that could also be called fixed intervals, i.e. the arcs that either correspond to activity duration, or **m**, **mi**, **f**, **fi**, **s** and **si** temporal logic relations; and
- weak intervals, that could also be called non-negative intervals, i.e. the arcs that correspond to basic constraints of the precedence method.

Two types of DGs are required to solve this problem: a forward pass DG that is used for calculating early start dates, and a backward pass DG for calculating latest dates. Figure 6 gives the forward pass DG that corresponds to example no. 1, and Figure 7 gives the backward pass DG of the same DS diagram. On both DGs, the dashed arc between FD_j'' and SD_k'' depicts a weak interval, while all other arcs are representing strong intervals.

Date calculations

The calculations of the dates of such a date graph are straightforward. This can be performed in three steps.

Step 1

The graph is broken down into sub-graphs by eliminating temporarily all weak arcs (cf. Figure 8). All the resulting sub-graphs should be irreflexive (self-loop free) and acyclic. A sub-graph G is said to be antecedent to a sub-graph G', if a weak interval, mapping from G to G' has been removed.

Step 2

Early date calculation (forward pass) starts in the sub-graphs that have no antecedent sub-graph, and within these sub-graphs, on the node that has no incoming arc. The early start date of the project ($ESD_{project}$) is assigned to these very first nodes. Formulas such as (1) are then used for propagating calculations in a given sub-graph.

Step 3

When all the dates are calculated in a given sub-graph, propagation can continue over weak intervals. To do that, CPM principles are applied. This is repeated until all the dates are calculated.

In a given sub-graph, backward calculations may sometimes be required for time stamping some of its nodes. This is the case for instance of ESD_k' of example no. 1.

The results of these forward pass calculations are earliest start or finish dates. A similar procedure running backward gives latest start or finish dates; the just calculated earliest finish date of the project ($EFD_{project}$) is assigned to the very last nodes, i.e. those with outgoing arcs. Critical paths and floats can be derived from this information.

Algorithms

The block diagram of Figure 9 gives an overview of the LDSM computational procedure. A more detailed view of the algorithms involved in the schedule calculation is given in the present section.

Seven algorithms need to be run successively to solve a LDS problem. A set of activities including their precedence and resources constraints is sufficient to describe the problem. Algorithm no. 1 aims at finding all the stations associated with the space-constrained activities. Algorithm no. 2 fragments the activity network into sub-activities. Algorithms nos. 3 and 4 generate the forward and backward pass DGs. Algorithms nos. 5 and 6 calculate, respectively, earliest and latest dates of sub-activities. Finally, algorithm no. 7 identifies the critical sub-activities.

For the sake of simplicity, these algorithms are presented in a pseudo-code form. Symbols $\wedge$ and $\vee$ stand respectively for 'and' and 'or'; statement 'a←b' means 'assign b to a'.

Let A be the set of all the activities of the project; each activity a_j is defined as in section 2. Let X be the set of all the stations x_j associated with this problem.

Algorithm no. 1: identification of the stations

10 $X \leftarrow \varnothing$
20 **for** $j = 1$ **to** $|A|$ **do:**
30 **if** a_j is a block activity **then:**
40 **if** $x_{j1} \notin X$ **then:** $X \leftarrow X \cup \{x_{j1}\}$ **end if**
50 **if** $x_{j2} \notin X$ **then:** $X \leftarrow X \cup \{x_{j2}\}$ **end if**
60 **end if**
70 $j \leftarrow j + 1$
80 **end do**
90 order X such as $x_p < x_{p+1}$, $1 \le p \le |X|$

Algorithm no. 2 provides a tool for fragmenting all the block activities into sub-activities. Bar activities are just duplicated into this new set. Let A' be the set of all the sub-activities of the project.

Note: The set Γ_j^{-1} of the predecessors of an activity a_j is made of 3-tuples made of:

- the predecessor activity featured by either a 6-tuple (block activity) or a 3-tuple (bar activity);
- the type of precedence constraint

$$\in \{ \underset{FS}{\prec}, \underset{SS}{\prec}, \underset{SF}{\prec}, \underset{FF}{\prec}, =, o \}$$

- the lag that separates the activities constrained.

Algorithm no. 2: fragmentation into sub-activities

10 $A' \leftarrow \varnothing$
20 order A such as $a_j \prec a_k$, $\forall a_j, a_k \in A \wedge j < k$
30 **for** $j = 1$ **to** $|A|$ **do:**
40 **if** a_j is a block activity $\wedge \theta_j > 0$ **then:**
50 **for** $n = k \mid x_k = x_{j1}$ **to** $\ell - 1 \mid x_\ell = x_{j2}$ **do:**
60 $P \leftarrow \begin{cases} \{ ((x_{n-1}, x_n, \theta_j, \mathsf{TS}_j, \Gamma_{n-1}^{-1}, \mathrm{r}_j), =, 0) \} \text{ if } (n > k) \wedge (\theta_j = \infty) \\ \{ ((x_{n-1}, x_n, \theta_j, \mathsf{TS}_j, \Gamma_{n-1}^{-1}, \mathrm{r}_j), o, \frac{x_n - x_{n-1}}{\theta_j}) \} \text{ if } (n > k) \wedge (\theta_j \neq \infty) \\ \{ (a_p, \sigma_{pj}, \mathsf{LAG}_{pj}) \mid a_p \in \Gamma_j^{-1} \wedge (a_p \text{ is a bar activity} \vee \\ \qquad (a_p \text{ is a block activity} \wedge]x_{j1}, x_{j2}[\cap]x_{p1}, x_{p2}[= \varnothing)) \} \text{ otherwise} \end{cases}$
70 $Q \leftarrow \begin{cases} \{ (s_q, \underset{FF}{\prec}, \mathsf{LAG}_{pj} + \mathsf{TS}_j) \mid s_q \in A' \wedge s_q \in a_p \in \Gamma_j^{-1} \wedge x_{q1} = n \wedge x_{q2} = n+1 \} \\ \qquad \text{if } 0 < \theta_q < \theta_j \\ \{ (s_q, \underset{SS}{\prec}, \mathsf{LAG}_{pj} + \mathsf{TS}_p) \mid s_q \in A' \wedge s_q \in a_p \in \Gamma_j^{-1} \wedge x_{q1} = n \wedge x_{q2} = n+1 \} \\ \qquad \text{if } \theta_j \le \theta_q \\ \{ (s_q, \underset{FS}{\prec}, \mathsf{LAG}_{pj}) \mid s_q \in A' \wedge s_q \in a_p \in \Gamma_j^{-1} \wedge x_{q1} = n \wedge x_{q2} = n+1 \} \\ \qquad \text{otherwise (i.e. if } \theta_q < 0) \end{cases}$
80 $A' \leftarrow A' \cup \{ (x_n, x_{n+1}, \theta_j, \mathsf{TS}_j, \Gamma_n^{-1} = P \cup Q, \mathrm{r}_j) \}$
90 $n \leftarrow n + 1$
100 **end do**
110 **else if** a_j is a block activity $\wedge \theta_j < 0$ **then:**
120 **for** $n = k \mid x_k = x_{j1}$ **to** $\ell + 1 \mid x_\ell = x_{j2}$ **do:**
130 $P \leftarrow \begin{cases} \{ ((x_{n+1}, x_n, \theta_j, \mathsf{TS}_j, \Gamma_{n+1}^{-1}, \mathrm{r}_j), o, \frac{x_n - x_{n+1}}{\theta_j}) \} \text{ if } n > k \\ \{ (a_p, \sigma_{pj}, \mathsf{LAG}_{pj}) \mid a_p \in \Gamma_j^{-1} \wedge a_p \text{ is a bar activity} \} \quad \text{otherwise} \end{cases}$
140 $Q \leftarrow \begin{cases} \{ (s_q, \underset{FF}{\prec}, \mathsf{LAG}_{pj} + \mathsf{TS}_j) \mid s_q \in A' \wedge s_q \in a_p \in \Gamma_j^{-1} \wedge x_{q1} = n \wedge x_{q2} = n-1 \} \\ \qquad \text{if } \theta_q \le \theta_j \\ \{ (s_q, \underset{SS}{\prec}, \mathsf{LAG}_{pj} + \mathsf{TS}_p) \mid s_q \in A' \wedge s_q \in a_p \in \Gamma_j^{-1} \wedge x_{q1} = n \wedge x_{q2} = n-1 \} \\ \qquad \text{if } \theta_j < \theta_q < 0 \\ \{ (s_q, \underset{FS}{\prec}, \mathsf{LAG}_{pj}) \mid s_q \in A' \wedge s_q \in a_p \in \Gamma_j^{-1} \wedge x_{q1} = n \wedge x_{q2} = n-1 \} \\ \qquad \text{otherwise (i.e. if } \theta_q > 0 \vee \theta_q = \infty) \end{cases}$
150 $A' \leftarrow A' \cup \{ (x_n, x_{n-1}, \theta_j, \mathsf{TS}_j, \Gamma_n^{-1} = P \cup Q, \mathrm{r}_j) \}$
160 $n \leftarrow n - 1$
170 **end do**
180 **else** (a_j is a bar activity)
190 $A' \leftarrow A' \cup \{ a_j \}$
200 **end if**
210 $j \leftarrow j + 1$
220 **end do**

Explanations

Line 30 scans all the activities of A in an ordered way given by line 20. For block activities, two cases are considered: either $\theta > 0$ (line 40) or $\theta < 0$ (line 110). Lines 50 and 120 scan in an ordered way all the stations spread all along the block activity that is being processed. This means that such an activity is broken down into as many sub-activities as there are intermediate stations between the activity's start and finish stations. P and Q are temporary sets that are used for identifying all the predecessors of the sub-activity that is scrutinized. The following rule is used for

transferring the set of predecessors of an activity a_j to its sub-activities:

- The sub-set of Γ_j^{-1} made of activities that are of bar type is transferred to the earliest of the sub-activities of a_j. This is the purpose of the last statement of lines 60 and 130.
- The sub-set of Γ_j^{-1} made of activities that are of block type is looked at through their corresponding sub-activities. A precedence constraint is set up between possible predecessor sub-activities and the sub-activity that is being processed if the start and end stations are matching. This is the purpose of the last statement of (lines 70 and 140). The first two lines of these statements aim at addressing the problem presented with example no. 2.

Once the predecessors of a sub-activity are defined, they are appended to the set A'.

Algorithms nos. 3 and 4 are used for generating the forward and backward pass DGs. Let G_F be the forward pass DG; G_F=<D_F; I_F> where D_F is the set of the earliest dates to be calculated and I_F is the set of 4-tuples featuring intervals. These 4-tuples are made of four data items: the extremity nodes/dates, the delay that separates these two dates and the type of interval that can be either strong or weak. Because precedence constraints can be of several types, the predecessor set Γ_j^{-1} of an activity a_j is made of triplets (as already seen) (a_k, σ_{kj}, LAG_{kj}), where a_k is an immediate predecessor of a_j, σ_{kj} denotes the type of precedence relation between a_k and a_j, and LAG_{kj} which is used to separate the finish of a_k and the start of a_j.

Algorithm no. 3: generation of the forward pass DG

10 $D_F \leftarrow \varnothing$
20 $I_F \leftarrow \varnothing$
30 **for** $j = 1$ **to** $|A'|$ **do:**
40 **if** $\mathsf{ESD}_j \notin D_F$ **then:** $D_F \leftarrow D_F \cup \{\mathsf{ESD}_j\}$ **end if**
50 **if** $\mathsf{EFD}_j \notin D_F$ **then:** $D_F \leftarrow D_F \cup \{\mathsf{EFD}_j\}$ **end if**
60 $I_F \leftarrow I_F \cup \left\{\left(\mathsf{ESD}_j, \mathsf{EFD}_j, \begin{cases} \frac{x_{j2}-x_{j1}}{\theta_j} + \mathsf{TS}_j \text{ if } a_j \text{ is a linear activity} \\ \mathsf{DUR}_j \text{ otherwise} \end{cases}, strong\right)\right\}$
70 **if** $\Gamma_j^{-1} \neq \varnothing$ **then:**
80 **for** $k = 1$ **to** $|\Gamma_j^{-1}|$ **do:**
90 **if** $\sigma_{kj} \subset \{\underset{FS}{\prec}, \underset{SS}{\prec}, \underset{SF}{\prec}, \underset{FF}{\prec}\}$ **then:**
100 **if** $(\mathsf{EFD}_k \notin D_F) \wedge (\sigma_{kj} \subset \{\underset{FS}{\prec}, \underset{FF}{\prec}\})$ **then:** $D_F \leftarrow D_F \cup \{\mathsf{EFD}_k\}$ **end if**
110 **if** $(\mathsf{ESD}_k \notin D_F) \wedge (\sigma_{kj} \subset \{\underset{SS}{\prec}, \underset{SF}{\prec}\})$ **then:** $D_F \leftarrow D_F \cup \{\mathsf{ESD}_k\}$ **end if**
120 $I_F \leftarrow I_F \cup \left\{\left(\begin{cases} \mathsf{ESD}_k \text{ if } \sigma_{kj} \subset \{\underset{FS}{\prec}, \underset{SF}{\prec}\} \\ \mathsf{EFD}_j \text{ if } \sigma_{kj} \subset \{\underset{FS}{\prec}, \underset{FF}{\prec}\} \end{cases}, \begin{cases} \mathsf{ESD}_k \text{ if } \sigma_{kj} \subset \{\underset{FS}{\prec}, \underset{SS}{\prec}\} \\ \mathsf{EFD}_j \text{ if } \sigma_{kj} \subset \{\underset{SF}{\prec}, \underset{FF}{\prec}\} \end{cases}, \mathsf{LAG}_{kj}, weak\right)\right\}$
130 **else if** $\sigma_{kj} \subset \{\mathbf{o}, =\}$ **then:**
140 **if** $\mathsf{ESD}_k \notin D_F$ **then:** $D_F \leftarrow D_F \cup \{\mathsf{ESD}_k\}$ **end if**
150 $I_F \leftarrow I_F \cup \left\{\left(\mathsf{ESD}_k, \mathsf{ESD}_j, \begin{cases} \frac{x_{k2}-x_{k1}}{\theta_k} \text{ if } \sigma_{kj} = \mathbf{o} \\ 0 \text{ otherwise} \end{cases}, strong\right)\right\}$
160 **end if**
170 $k \leftarrow k+1$
180 **end do**
190 **end if**
200 $j \leftarrow j+1$
210 **end do**

Explanations

Line 30 aims at scanning all the sub-activities of A'. The duration of a sub-activity is assigned into the DG in line 60. Line 80 scans all the predecessors of the sub-activity being scrutinized. If the constraint that links activities a_k and a_j is a weak constraint, the corresponding interval is appended to I_F as from line 120; if it is a strong constraint, it is assigned to I_F as from line 150. Let G_B be the backward pass DG; G_B=<D_B; I_B>. The generation of this backward pass DG is identical to the one of the forward pass DG given by algorithm no. 3, except the replacement of F indices by B indices, Γ_j^{-1} by Γ_j and the replacement of lines 140 and 150 by:

...
140 **if** $\mathsf{EFD}_k \notin D_B$ **then:** $D_B \leftarrow D_B \cup \{\mathsf{EFD}_k\}$ **end if**
150 $I_B \leftarrow I_B \cup \left\{\left(\mathsf{EFD}_k, \mathsf{EFD}_j, \begin{cases} \frac{x_{j2}-x_{j1}}{\theta_j} \text{ if } \sigma_{kj} = \mathbf{o} \\ 0 \text{ otherwise} \end{cases}, strong\right)\right\}$
...

Algorithms 5 and 6 provide a way for calculating a date, earliest dates (ED_j) for the vertices of the forward pass DG; latest dates (LD_j) for the vertices of the backward pass DG. Let $\mathscr{P}_A$ be the set of the activities that are scheduled at a given date, and $\mathscr{E}_A$ the set of the activities that are eligible for being scheduled (mainly because their predecessor activities have been scheduled) at that same date.

Algorithm no. 5: calculation of the earliest dates

10 $\mathscr{P}_A \leftarrow \varnothing$
20 **repeat:**
30 $\mathscr{E}_A \leftarrow \{a_p \mid a_p \in A \wedge \Gamma_p^{-1} \subseteq \mathscr{P}_A \wedge a_p \notin \mathscr{P}_A\}$
40 $\mathscr{P}_W \leftarrow \varnothing$
50 **if** $\Gamma_p^{-1} = \varnothing$ **then:**
60 $\mathscr{E}_D \leftarrow \{d_j \mid d_j \in a_p \in \mathscr{E}_A \wedge \Gamma_j^{-1} = \varnothing\}$
70 $\mathsf{ED}_j = 0$ or the project start date
80 $\mathscr{P}_W \leftarrow \mathscr{P}_W \cup \{d_j\}$
} $\forall d_j \in \mathscr{E}_D$
90 **else** $(\Gamma_p^{-1} \neq \varnothing)$
100 $\mathscr{E}_D \leftarrow \{d_j \mid d_j \in a_p \in \mathscr{E}_A \wedge u_{kj} \in I_F \wedge u_{kj} \text{ is weak}\}$
110 $\mathsf{ED}_j = \max_{d_k \in \Gamma^{-1}_{\text{weak } j}} (\mathsf{ED}_k + \mathsf{LAG}_{kj})$
120 $\mathscr{P}_W \leftarrow \mathscr{P}_W \cup \{d_j\}$
} $\forall d_j \in \mathscr{E}_D$
130 **end if**
140 $\mathscr{P}_S \leftarrow \{d_j \mid d_j \in \mathscr{P}_W \wedge \mathsf{ED}_j = \min_{d_k \in \mathscr{P}_W} (\mathsf{ED}_k)\}$
150 **run** SubNetPropagationFW(a_p)
} $\forall a_p \in \mathscr{E}_A$
160 $\mathscr{P}_A \leftarrow \mathscr{P}_A \cup \mathscr{E}_A$
170 **while** $|\mathscr{P}_A| < |A|$

```
200  procedure SubNetPropagationFW(a_p)
210    repeat:
220      E_F ← { d_j | u_kj ∈ I_F ∧ u_kj is strong ∧ d_k ∈ P_S ∧ d_j ∉ P_S }
230      if d_j ∈ P_W ∧ ED_k + LAG_kj < ED_j then :
240        P_S ← { d_j }
250        run SubNetPropagationFW(a_p)                    } ∀d_j ∈ E_F
260      else
270        ED_j ← ED_k + LAG_kj
280      endif
290      E_B ← { d_j | u_jk ∈ I_F ∧ u_jk is strong ∧ d_k ∈ P_S ∧ d_j ∉ P_S }
300      if d_j ∈ P_W ∧ ED_k − LAG_jk < ED_j then :
310        P_S ← { d_j }
320        run SubNetPropagationFW(a_p)                    } ∀d_j ∈ E_B
330      else
340        ED_j ← ED_k − LAG_jk
350      endif
360      P_S ← P_S ∪ E_F ∪ E_B
370    while |P_S| < |a_p|
```

Explanations

The set $\mathscr{P}_A$ of the activities, i.e. of the sub-graphs, that have already been scheduled, is initialized in line 10. A repeat…while statement (lines 20 and 170) is used for scanning all the activities of A.

The set $\mathscr{E}_A$ of the eligible activities is set up in line 30. It is made of the activities of A that have not yet been scheduled, i.e., $a_p \notin \mathscr{P}_A$ but that do have their predecessors scheduled, i.e. $\Gamma_p^{-1} \subseteq \mathscr{P}_A$.

The set $\mathscr{P}_A$ is initialized to empty set with line 40, and reinitialized with every new activity being scrutinized. This set is used as a starting point for date calculations within a given activity.

Two cases may occur for date eligibility, i.e. for identification of the vertices of the forward pass DG that can be calculated. Either the activity has no predecessor (line 50) or it has one (line 90). For both cases, a set $\mathscr{E}_D$ of eligible dates is set up (lines 60 and 100). This set is made of the date vertices that either

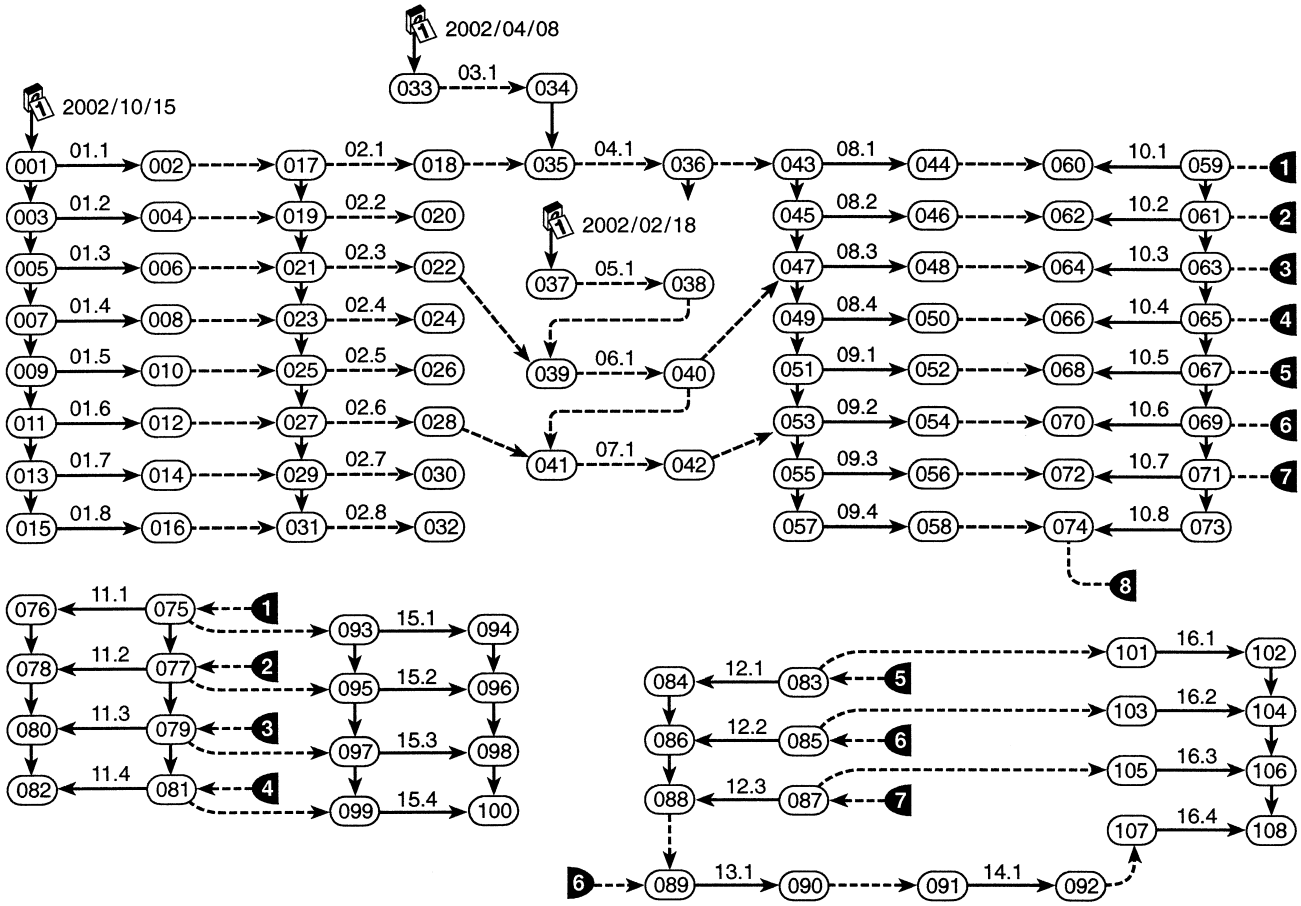

Figure 13 Example-backward pass DG

act. no.	Sub-act.	Vertex	Earliest start date ESD_j	Latest start date LSD_j	Vertex	Earliest finish date EFD_j	Latest finish date LFD_j	Total float TF_j
01	01.1	003	Mon 2001/10/15	Mon 2001/12/03	002	Fri 2002/01/04	Fri 2002/02/22	7w.
	01.2	003	Mon 2001/10/15	Mon 2001/12/03	004	Fri 2002/01/04	Fri 2002/02/22	7w.
	01.3	005	Mon 2001/10/15	Mon 2001/12/03	006	Fri 2002/01/04	Fri 2002/02/22	7w.
	01.4	007	Mon 2001/10/15	Mon 2001/12/03	008	Fri 2002/01/04	Fri 2002/02/22	7w.
	01.5	009	Mon 2001/10/15	Mon 2001/12/03	010	Fri 2002/01/04	Fri 2002/02/22	7w.
	01.6	011	Mon 2001/10/15	Mon 2001/12/03	012	Fri 2002/01/04	Fri 2002/02/22	7w.
	01.7	013	Mon 2001/10/15	Mon 2001/12/03	014	Fri 2002/01/04	Fri 2002/02/22	7w.
	01.8	015	Mon 2001/10/15	Mon 2001/12/03	016	Fri 2002/01/04	Fri 2002/02/22	7w.
02	02.1	017	Mon 2002/01/07	Mon 2001/02/25	018	Fri 2002/03/15	Fri 2002/05/03	7w.
	02.2	019	Mon 2002/01/07	Mon 2002/02/25	020	Fri 2002/03/15	Fri 2002/05/03	7w.
	02.3	021	Mon 2002/01/07	Mon 2002/02/25	022	Fri 2002/03/15	Fri 2002/05/03	7w.
	02.4	023	Mon 2002/01/07	Mon 2002/02/25	024	Fri 2002/03/15	Fri 2002/05/03	7w.
	02.5	025	Mon 2002/01/07	Mon 2002/02/25	026	Fri 2002/03/15	Fri 2002/05/03	7w.
	02.6	027	Mon 2002/01/07	Mon 2002/02/25	028	Fri 2002/03/15	Fri 2002/05/03	7w.
	02.7	029	Mon 2002/01/07	Mon 2002/02/25	030	Fri 2002/03/15	Fri 2002/05/03	7w.
	02.8	031	Mon 2002/01/07	Mon 2002/02/25	032	Fri 2002/03/15	Fri 2002/05/03	7w.
03	03.1	033	Mon 2002/04/08	Mon 2002/04/08	034	Fri 2002/05/03	Fri 2002/05/03	0w.
04	04.1	035	Mon 2002/05/06	Mon 2002/05/06	036	Fri 2002/05/17	Fri 2002/05/17	0w.
05	05.1	037	Mon 2002/02/18	Mon 2002/03/20	038	Fri 2002/04/12	Fri 2002/05/14	4w.
06	06.1	039	Mon 2002/04/15	Mon 2002/05/15	040	Fri 2002/05/10	Fri 2002/06/11	4w.
07	07.1	041	Mon 2002/05/13	Mon 2002/06/24	042	Fri 2002/06/07	Fri 2002/07/19	6w.
08	08.1	043	Mon 2002/05/20	Mon 2002/05/20	044	Fri 2002/05/31	Fri 2002/05/31	0w.
	08.2	045	Mon 2002/05/20	Mon 2002/05/20	046	Tue 2002/06/25	Tue 2002/06/25	0w.
	08.3	047	Mon 2002/06/12	Mon 2002/06/12	048	Tue 2002/06/25	Tue 2002/06/25	0w.
	08.4	049	Mon 2002/06/12	Mon 2002/06/12	050	Tue 2002/07/19	Tue 2002/07/19	0w.
09	09.1	051	Mon 2002/05/15	Mon 2002/06/26	052	Fri 2002/06/21	Fri 2002/08/02	6w.
	09.2	053	Mon 2002/06/10	Mon 2002/07/22	054	Fri 2002/06/21	Fri 2002/08/02	6w.
	09.3	055	Mon 2002/06/10	Mon 2002/07/22	056	Tue 2002/07/16	Tue 2002/08/27	6w.
	09.4	057	Mon 2002/07/03	Mon 2001/08/14	058	Wed 2002/07/24	Wed 2002/09/03	6w.

Figure 14 Example-results of the computations (1/2)

have no in-coming edge (case $\Gamma_p^{-1}=\emptyset$) or that have weak type in-coming edges (case $\Gamma_p^{-1}\neq\emptyset$). The project start date is assigned to the earliest date ED_j if the parent activity has no predecessor (line 70); otherwise, it is the latest date among those associated to the date vertices of Γ_j^{-1} that is assigned to ED_j (line 110). The vertex d_j is then appended to the set $\mathscr{P}_W$ (lines 80 and 120).

The date propagation within a given activity/sub-graph requires a unique starting point; this is the purpose of line 140. When $\mathscr{P}_W$ is made of more than one date, it is the earliest one that is preferred and appended to $\mathscr{P}_S$.

The procedure SubNetPropagationFW is called for propagating the dates of a given activity/sub-graph (line 150). The sets $\mathscr{E}_F$ and $\mathscr{E}_B$ are used for identifying eligible dates (lines 220 and 290), i.e. the dates that are not yet scheduled ($d_j \notin \mathscr{P}_S$) but have adjacent vertices d_k already scheduled ($d_k \notin \mathscr{P}_S$). It may happen that such a date has already been scheduled: $d_j \in \mathscr{P}_W$ but $d_j \notin \mathscr{P}_S$. If the newly calculated ED_j is greater than the previously calculated (lines 230 and 300), this latter is superseded by the new one (lines 240 and 310). Otherwise, the procedure SubNetPropagationFW is run again with a new starting point.

Algorithm no. 6: calculation of the latest dates

10 $\mathscr{P}_A \leftarrow \emptyset$

20 **repeat:**

30 $\mathscr{E}_A \leftarrow \{ a_p \mid a_p \in A \wedge \Gamma_p \subseteq \mathscr{P}_A \wedge a_p \notin \mathscr{P}_A \}$

40 $\mathscr{P}_W \leftarrow \emptyset$

50 **if** $\Gamma_p = \emptyset$ **then:**

```
60    E_D ← { d_j | d_j ∈ a_p ∈ E_A ∧ Γ_j = ∅ }
70    LD_j = max_{d_k ∈ D_F} (ED_k) i.e. the project finish date
80    P_W ← P_W ∪ { d_j }                                        } ∀d_j ∈ E_D
90  else (Γ_p ≠ ∅)
100   E_D ← { d_j | d_j ∈ a_p ∈ E_A ∧ u_jk ∈ I_B ∧ u_jk is weak }
110   LD_j = min_{d_k ∈ Γ_weak j} (LD_k − LAG_jk)
120   P_W ← P_W ∪ { d_j }                                        } ∀d_j ∈ E_D
130 end if
140 P_S ← { d_j | d_j ∈ P_W ∧ LD_j = max_{d_k ∈ P_W} (LD_k) }    } ∀a_p ∈ E_A
150 run SubNetPropagationBW(a_p)
160 P_A ← P_A ∪ E_A
170 while |P_A| < |A|
200 procedure SubNetPropagationBW(a_p)
210   repeat:
220     E_F ← { d_j | u_kj ∈ I_B ∧ u_kj is strong ∧ d_k ∈ P_S ∧ d_j ∉ P_S }
230     if d_j ∈ P_W ∧ LD_k + LAG_kj > LD_j then :
240       P_S ← { d_j }
250       run SubNetPropagationBW(a_p)
260     else
270       LD_j ← LD_k + LAG_kj                                   } ∀d_j ∈ E_F
280     end if
290     E_B ← { d_j | u_jk ∈ I_B ∧ u_jk is strong ∧ d_k ∈ P_S ∧ d_j ∉ P_S }
300     if d_j ∈ P_W ∧ LD_k − LAG_jk > LD_j then :
310       P_S ← { d_j }
320       run SubNetPropagationBW(a_p)
330     else
340       LD_j ← LD_k − LAG_jk                                   } ∀d_j ∈ E_B
350     end if
360     P_S ← P_S ∪ E_F ∪ E_B
370   while |P_S| < |a_p|
```

Finally, algorithm no. 7 calculates the critical activities of the project. The critical activities of an activity network are defined as activities for which earliest dates and latest dates coincide. This means that one cannot delay one of these activities without delaying proportionally the completion date of the project.

```
10  d_p ← d_j | d_j ∈ a_p ∧ ED_j = min_{d_k ∈ a_p} (ED_k)      } ∀a_p ∈ A
20  TF_p ← LD_p − ED_p
30  A_C ← { a_p | a_p ∈ A ∧ TF_p = 0 }
```

Explanations

Line 10 picks up a date d_p, the earliest, among all the dates of an activity a_p. The total float TF_p is calculated as the difference between the latest and the earliest dates that have been calculated for d_p (line 20). The set AC of the critical activities of the project is made of all the activities that have a total float equal to zero.

Example

A wide range of projects use linear scheduling approaches: construction of highways and railways, repavement of runways, excavation of tunnels, installation of pipelines, etc. The construction of large particle accelerators also involves such scheduling approaches. The LHC (Large Hadron Collider) that is being built near Geneva, Switzerland, uses at co-ordination level a scheduling approach that is very similar to the LDSM. The one used for that project is slightly more complex because it also integrates a line-of-balance mechanism.

The LHC will provide particle physics community with a tool to reach the energy frontier above 1 TeV. To deliver 14 TeV proton-proton collisions, it will operate with about 1700 cryo-magnets using NbTi superconductors cooled at 1.8 K. These cryo-magnets will be installed in the 27-km long, 100-m underground ring tunnel that was excavated 15 years ago for housing the LEP (Large Electron-Positron) accelerator. After a decade of research and development, the LHC main components are being manufactured in industry.

The installation works have started with the refurbishing of the existing infrastructures. The installation schedule of the LHC project consists of about 2000 work units. The construction of new civil works has started in 1998. The particle physics community expects to have this new accelerator installed and fully commissioned by mid-2007. For the sake of simplicity, we have limited our example to a small sub-set of this large-scale project: the installation of the general services of sectors 7–8 (one-eighth of the main ring). The LHC co-ordination schedule can be seen from www.cern.ch. A summary of this schedule is provided in the Appendix of this paper.

The sub-set used as an example is made of 16 activities, or work units, as referred to in the LHC project jargon. These activities are described in Figure 10, including their duration and predecessors.

Activities 03 and 05 are carried out away from the installation site (i.e. the LHC tunnel). For that very reason, these activities are of bar type. Activities 01 and 02 are activities that require the whole tunnel for being performed; they are of discrete space-constrained type for featuring this characteristic. Activities 04, 06, 07, 13 and 14 are spot activities. They correspond to very specific works to be performed in punctual locations in the LHC tunnel; few meter lengths of tunnel are concerned. All other activities are linear

act. no.	Sub-act.	Vertex	Earliest start date ESD_j	Latest start date LSD_j	Vertex	Earliest finish date EFD_j	Latest finish date LFD_j	Total float TF_j
10	10.1	059	Wed 2002/06/12	Wed 2002/06/12	060	Wed 2002/07/03	Wed 2002/07/03	0w.
	10.2	061	Thu 2002/06/13	Thu 2002/06/13	062	Fri 2002/07/19	Fri 2002/07/19	0w.
	10.3	063	Mon 2002/07/01	Mon 2002/07/01	064	Fri 2002/07/19	Fri 2002/07/19	0w.
	10.4	065	Mon 2002/07/01	Mon 2002/07/01	066	Fri 2002/08/09	Fri 2002/08/09	0w.
	10.5	067	Mon 2002/07/22	Mon 2002/07/22	068	Fri 2002/08/30	Fri 2002/08/30	0w.
	10.6	069	Mon 2002/08/12	Mon 2002/08/12	070	Fri 2002/08/30	Fri 2002/08/30	0w.
	10.7	071	Mon 2002/08/12	Mon 2002/08/12	072	Wed 2002/09/18	Wed 2002/09/18	0w.
	10.8	073	Thu 2002/08/29	Thu 2002/08/29	074	Tue 2002/09/24	Tue 2002/09/24	0w.
11	11.1	075	Wed 2002/07/03	Mon 2002/08/26	076	Wed 2002/07/17	Mon 2002/09/09	8w.
	11.2	077	Thu 2002/07/04	Tue 2002/08/27	078	Mon 2002/08/12	Fri 2002/10/04	8w.
	11.3	079	Tue 2002/07/30	Mon 2002/09/23	080	Mon 2002/08/12	Fri 2002/10/04	8w.
	11.4	081	Tue 2002/07/30	Mon 2002/09/23	082	Tue 2002/09/10	Fri 2002/11/01	8w.
12	12.1	083	Mon 2002/08/12	Mon 2002/08/12	084	Fri 2002/09/20	Fri 2002/09/20	0w.
	12.2	085	Mon 2002/09/09	Mon 2002/09/09	086	Fri 2002/09/20	Fri 2002/09/20	0w.
	12.3	087	Mon 2002/09/09	Mon 2002/09/09	088	Fri 2002/10/18	Fri 2002/10/18	0w.
13	13.1	089	Mon 2002/10/21	Mon 2002/10/28	090	Fri 2002/11/08	Fri 2002/11/15	1w.
14	14.1	091	Mon 2002/11/11	Mon 2002/11/18	092	Fri 2002/11/22	Fri 2002/11/29	1w.
15	15.1	093	Wed 2002/07/17	Wed 2002/09/09	094	Wed 2002/08/14	Mon 2002/10/07	8w.
	15.2	095	Thu 2002/07/18	Tue 2002/09/10	096	Fri 2002/09/27	Fri 2002/11/22	8w.
	15.3	097	Mon 2002/09/02	Mon 2002/10/28	098	Mon 2002/09/30	Fri 2002/11/22	8w.
	15.4	099	Tue 2002/09/03	Mon 2002/10/28	100	Tue 2002/11/19	Fri 2002/01/10	8w.
16	16.1	101	Mon 2002/08/26	Mon 2002/08/26	102	Mon 2002/11/11	Mon 2002/11/11	0w.
	16.2	103	Tue 2002/10/15	Tue 2002/10/15	104	Mon 2002/11/11	Mon 2002/11/11	0w.
	16.3	105	Tue 2002/10/15	Tue 2002/10/15	106	Fri 2002/12/17	Fri 2002/12/17	0w.
	16.4	107	Mon 2002/12/02	Mon 2002/12/02	108	Fri 2002/01/10	Fri 2002/01/10	0w.

Figure 15 Example-results of the computations (2/2)

space-constrained activities; all their θ_j have been set to positive values in order to provide a pre-optimization of the schedule.

The seven algorithms presented in the previous section have been run on this sub-set of activities; the results of the computations are given in Figures 11–17. The result of algorithm no. 2, i.e. the corresponding sub-activities is given in Fig. 11. The forward and backward pass DGs, as generated using algorithms 3 and 4, are given in Figures 12 and 13. The results of the computations obtained using algorithms 5, 6 and 7 are given in Figures 14 and 15. Two Gantt charts are proposed, one featuring the earliest dates for these 16 activities (Figure 16), another with the latest dates (Figure 17).

Based on this case experience, this approach improves both the analysis of the activity network and, because of a better visualization, ease the communication of the schedule. Rescheduling and the dynamic impact of changes can be better understood.

Conclusion

For some specific types of construction projects, specifically the ones that require activities with linear development and/or repetitive activities, the classical project scheduling approaches, based on the CPM or PDM, are not the most suitable. Therefore, few specific approaches were developed to cope with these types of projects; but none of them addresses specifically the mixture of linear type activities together with non-space-constrained activities, in a single resource-driven scheduling system.

With the linear-discrete scheduling model, the linear problem is transformed into a standard resource-contrained project scheduling problem, onto which a wide range of algorithms can be applied, especially those for optimizing the allocation of resources (see, for example, Hartmann, 2001), including the critical chain approach (Newbold, 1998).

One of the key features of the methodology is the implementation of a sub-set of Allen's temporal

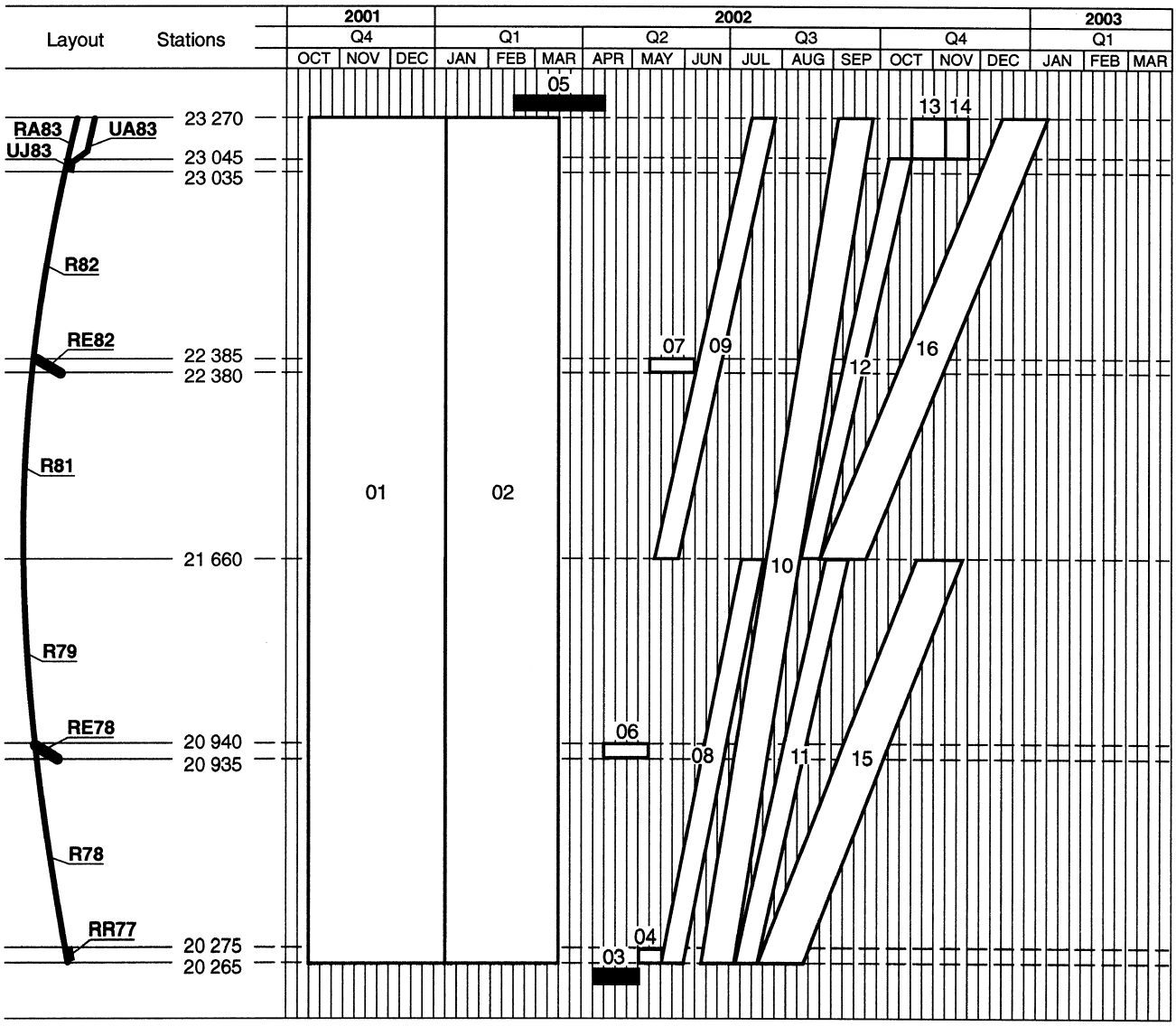

Figure 16 Example-Gantt chart with earliest dates

logic relations, used to avoid adverse discontinuities. If we look at the past, activity-on-arrow diagrams were first proposed in the late 1950s, for finding solutions to project activity networks. These networks could model only finish–start precedence constraints. Activity-on-node diagrams offer more flexibility from that point of view: in addition to FS precedence constraints, the project scheduler has the possibility to model SS, SF or FF constraints, with or without lag. The Allen's temporal logic relations offer 13 additional configurations that can be very useful to project schedulers, especially to those who are in charge of large-scale speculative project schedules, and who have not the possibility to describe thousand of activities. The scheduling of space-constrained projects are among the ones that can benefit from using precedence constraints mixed with Allen's relations, but such a scheduling approach can be beneficial to many other project contexts.

Such an integrated method provides project management practitioners with better means to handle time variation in their tasks. In doing this, it enables project managers to better master overall time usage in the project while at the same time giving them better control to meet project milestones and, most importantly, to reduce project lead-times. Lead-time reduction correlates with a better quality, a better productivity, and by the way to projects with higher return on investment.

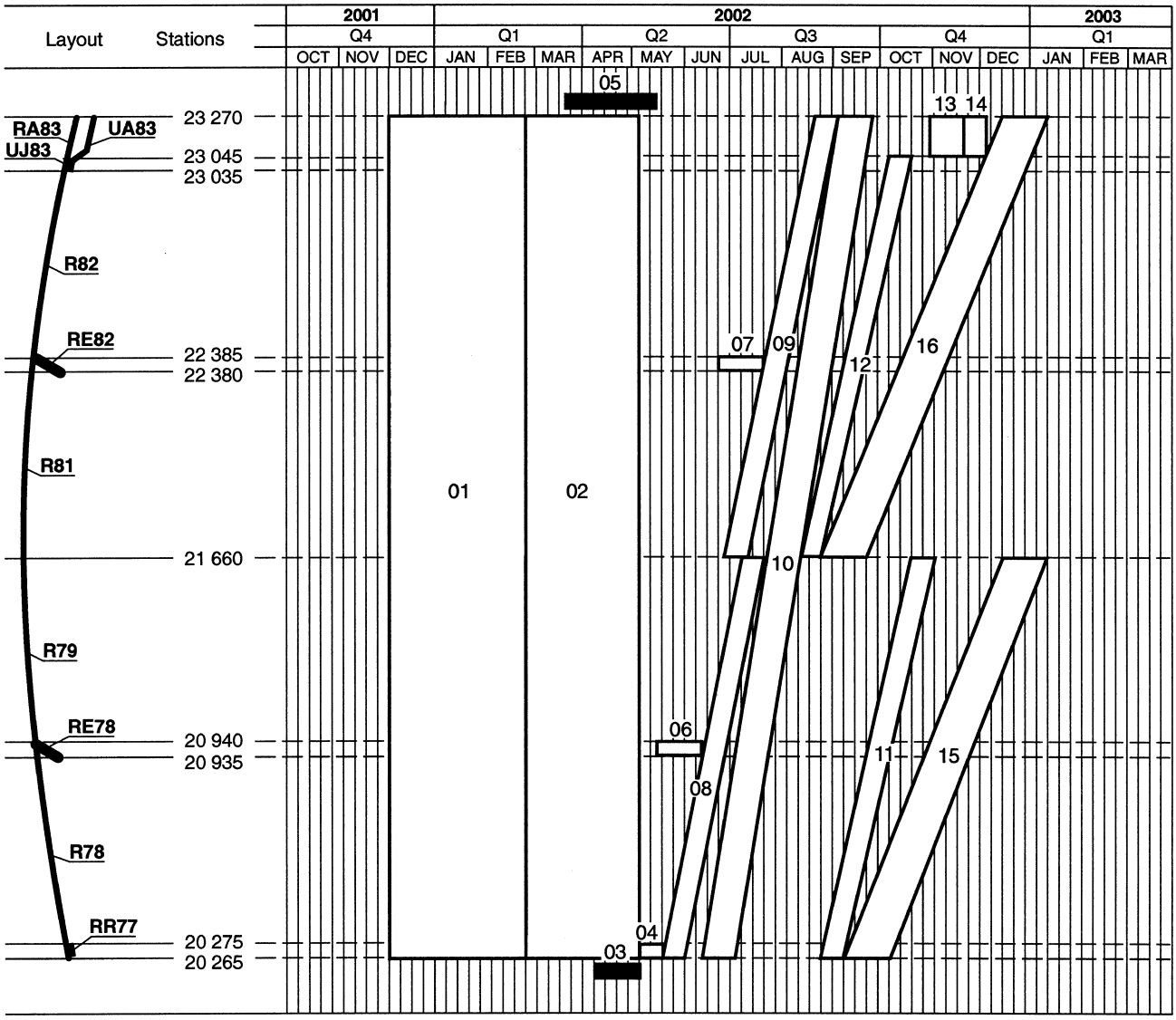

Figure 17 Example-Gantt chart with latest dates

References

Allen, J.F. (1983) Maintaining knowledge about temporal intervals. *Communications of the ACM*, **26**(11), 832–43.

Ashley, D.B. (1980) Simulation of repetitive-unit construction. *Journal of Construction Division, ASCE*, **106**(2), 185–94.

Chrzanowski, E.N. and Johnston, D.W. (1986) Application of linear scheduling. *Journal of Construction Engineering and Management, ASCE*, **112**(4), 476–91.

Eldin, N.N. and Senouci, A.B. (1994) Scheduling and control of linear projects. *Canadian Journal of Civil Engineering*, **21**(2), 219–30.

Eldin, N.N. and Senouci, A.B. (2000) Scheduling of linear projects with single loop structures. *Advances in Engineering Software*, **31**, 803–14.

El-Rayes, K. (2001) Object oriented model for repetitive construction scheduling. *Journal of Construction Engineering and Management, ASCE*, **127**(3), 199–205.

El-Rayes, K. and Moselhi, O. (1998) Resource-driven scheduling of repetitive activities. *Construction Management and Economics*, **16**(4), 433–46.

El-Rayes, K. and Moselhi, O. (2001) Optimizing resource utilization for repetitive construction projects. *Journal of Construction Engineering and Management, ASCE*, **127**(1), 18–27.

El-Rayes, K., Ramanathan, R. and Moselhi, O. (2002) An object-oriented model for planning and control of housing construction. *Construction Management and Economics*, **20**(3), 201–10.

Handa, V. and Barcia, R. (1986) Linear scheduling using optimal control theory. *Journal of Construction Engineering and Management, ASCE*, **112**(3), 387–93.

Harmelink, D.J. (2001) Linear scheduling model: float characteristics. *Journal of Construction Engineering and Management, ASCE*, **127**(4), 255–60.

Harmelink, D.J. and Rowings, J.E. (1998) Linear scheduling model: development of controlling activity path. *Journal of Construction Engineering and Management, ASCE*, **124**(4), 263–8.

Hartmann, S. (2001) *Project scheduling under limited resources: models, methods and applications*, Springer, Berlin.

Johnston, D.W. (1981) Linear scheduling method for highway construction. *Journal of Construction Division, ASCE*, **107**(2), 247–61.

Kavanagh, D.P. (1985) SIREN: a repetitive construction simulation model. *Journal of Construction Engineering and Management, ASCE*, **111**(3), 308–23.

Leu, S. and Hwang, S. (2001) Optimal repetitive scheduling model with shareable resource constraint. *Journal of Construction Engineering and Management, ASCE*, **127**(4), 270–80.

Moselhi, O. and El-Rayes, K. (1993) Scheduling of repetitive projects with cost optimization. *Journal of Construction Engineering and Management, ASCE*, **119**(4), 681–97.

Moselhi, O. and Hassanein, A. (2003) Optimized scheduling of linear projects. *Journal of Construction Engineering and Management, ASCE*, **129**(6), 664–73.

Newbold, R.C. (1998) *Project Management in the Fast Lane, Applying the Theory of Constraints*, St-Lucie Press, Boca Raton, FL.

O'Brien, J.J., Kreitzberg, F.C. and Mikes, F.W. (1985) Network scheduling variation for repetitive work. *Journal of Construction Engineering and Management, ASCE*, **111**(2), 105–16.

Reda, R. (1990) RPM: repetitive project modeling. *Journal of Construction Engineering and Management, ASCE*, **116**(2), 316–30.

Russell, A.D. and Caselton, W.F. (1988) Extensions to linear scheduling optimization. *Journal of Construction Division, ASCE*, **114**(1), 36–52.

Russell, A.D. and Wong, W.C.M. (1993) New generation of planning structures. *Journal of Construction Engineering and Management, ASCE*, **119**(2), 196–214.

Selinger, S. (1980) Construction planning for linear projects. *Journal of Construction Division, ASCE*, **106**(2), 195–205.

Suhail, S.A. and Neale, R.H. (1994) CPM/LOB: new methodology to integrate CPM and Line of Balance. *Journal of Construction Engineering and Management, ASCE*, **120**(2), 667–84.

Zaidi, A.K. and Levis, A.H. (2001) TEMPER: a temporal programmer for time-sensitive control of discrete events. *IEEE Transaction on Systems, Man & Cybernetics – Part A*, **31**(6), 485–96.