Scheduling of Repetitive Activities for Height-Rise Buildings: Optimisation by Genetic Algorithms

Prof. Mohammed ALJOMA, Ph.D.

Construction Management Department, Aleppo University, Aleppo, Syria.
E-mail: mraljoma@hotmail.com

ABSTRACT

In this paper, a developed prototype for the scheduling of repetitive activities in height-rise buildings was presented. The activities that describe the behaviour of the most of activities in multi-storey buildings are scheduled using the developed approach. The prototype combines three methods to attain the optimized planning. The methods include Critical Path Method (CPM), Gantt and Line of Balance (LOB). The developed prototype; POTER is used to schedule repetitive and non-repetitive activities with respect to all constraints that can be automatically generated using a generic database. The prototype uses the method of genetic algorithms for optimising the planning process. As a result, this approach enables contracting organisations to evaluate various planning solutions that are calculated, tested and classified by POTER to attain an optimal time-cost equilibrium according to their own criteria of time or coast.

Keywords
planning scheduling; genetic algorithms; repetitive activity; project duration

Acknowledgements

Construction planning involves the selection of proper methods, crew sizes, equipments and technologies to perform the tasks required in construction projects. Management project engineers meanwhile are faced with repetitive activities in multi-storey buildings involving similar unities of work. The planning of repetitive activities is based on the best possible organisation of available team crews in work sites, wherein the number of employees is limited and additional workers are very expensive.

The organisation of repetitive activities was always researched, ever since man began to construct buildings in cities. The sequencing of these activities is controlled by constraints between activities executed in the same unity or in different unities. To incorporate these constraints in the schedule, the scheduling process must ensure the constraints are satisfactory for all activities on every floor [20].

The planning of repetitive activities has used different patterns depending on the intended method. The Line-Of-Balance (LOB) is adopted for linear scheduling [16] and [10] but it is not very effective when there are non-repetitive activities. Other methods have used Critical Path Method with LOB to solve this problem [17]. The goal of these studies was not only to make a planning of repetitive activities in order to minimize the project duration but also to reduce its cost. Although, the optimisation of two criteria was developed for planning project, little or non-reported researches focusing on optimisation project duration and cost using repetitive activities aspects and genetic algorithms techniques were boarded.

The exploitation of optimisation by genetic algorithms has great applications after the high processors revolutions that reduced the time needed to scan the possible solutions to find the best possible solution. The GA method nowadays is very popular as an optimisation method. Many researches based on this method were established.[2], [3], [6], [11].

Objectives

This paper describes two parts. A structured procedure to incorporate succession and imposed constrains validation to schedule repetitive work in height-rise building. Our approach proposes the use of previous methods, PERT, Gantt and LOB, to make a general and simpler model that meets the need of validating the different constraints for repetitive and non-repetitive activities. Firstly, the model was developed in order to construct a project planning from the use of a list of stocked cases. Theses cases consider all possible constraints in order to calculate the project duration. Secondly, it takes into account different types of repetitive activities that have either upwards or downwards behaviour as well as non-repetitive activities. They can be seen clearly in the construction and final finishing activities. Finally, it describes and validates imposed and succession constraints that could join the activities in height-rise buildings projects. The model is able to construct a project scheduling by calculating both start and end times of each activity according to the set of constraints, activity duration and allocated teams.

The second part of our approach describes the utilisation of genetic algorithms to optimise the management repetitive activities scheduling. GA is used to find the shortest schedule duration beyond the large combinational problem that describes the possible sequences of activities and team organizing.

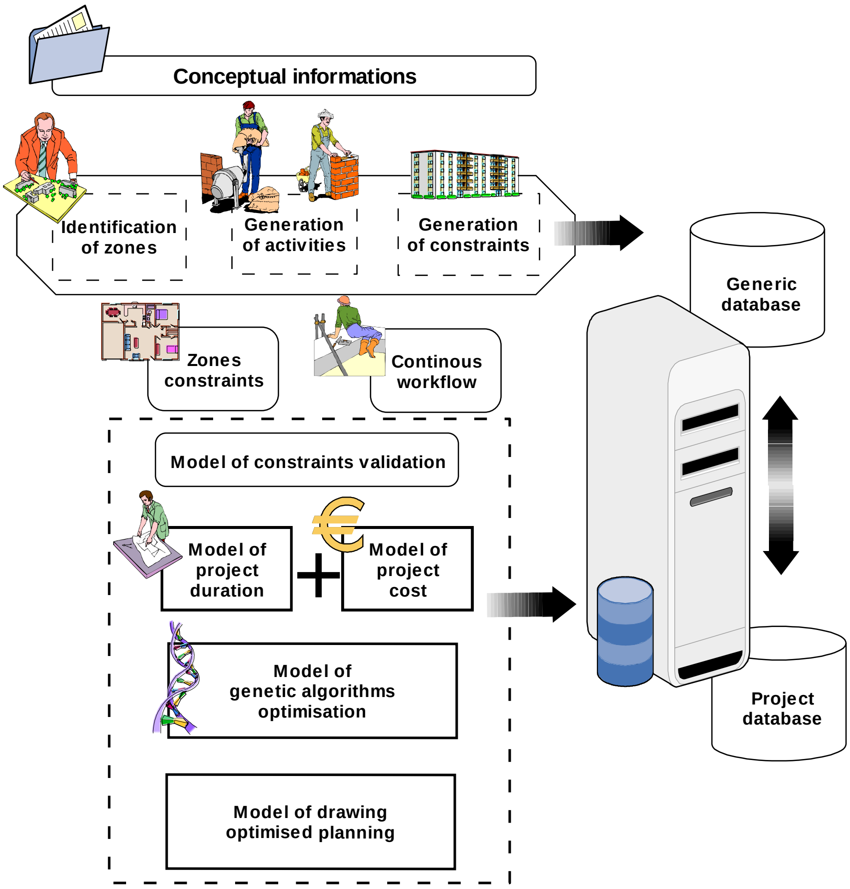


Figure 1: POTER prototype

Our programmed approach (POTER) is composed of a database and models that helps to create and calculate the project duration and cost (Figure 1).

• Generic database: All information of activities and their options is stocked. It contains the description of each task, level, duration, cost, etc.

• Project database: All project information that concern the current project can be either generated by the generic database or by the user.

These two databases are supplied by the technical files of the construction company, i.e. results, queries and other interesting information that can be used for new projects. Our prototype contains different models.

1. The model of zones identification: Typical project zones such as buildings, floors and apartments are generated automatically. Otherwise the user enters them manually. All project units such as buildings, divisions and locals are taken into account.

2. The model of activities and constraints generation: These two models carry out the tasks that should be realized in a project. The activities and units are generated according to realization options. The relations between units and activities are automatically identified as constraints; i. e. the slab of next floor cannot be done before beams or columns of current floor. Other special constraints should be entered manually by the user.

3. The optimisation model where three sub-models are embedded: Optimisation model using genetic algorithms, project cost model and project duration model. Taking into account the large number of planning solutions that give different times and costs, this model gives a solution that can be optimal or near-optimal. The best solution will be scheduled and its resource results classified in the final step of the optimisation.

4. Plotting results: this procedure use the list of optimised activities that include their durations, their constraints, their sequences, their zones and their crew teams to drew all the activities in the scale of time whatever the criteria of optimisation.

Division of the project

Schedule planning begins by breaking down the project into physical sections. A section is usually defined as the smallest area where the framework of the building can be erected independent of other sections (Figure 2). This means that the sections are alike for every floor. Sections are further divided into floors and/or spaces. The place hierarchy is project-specific: different division principles can be used in different projects.

Section division allows interior works to begin earlier in the first section and thus reduces the project duration. The schedule can be compressed by choosing the optimal construction sequence. Schedule compression requires that the same construction order and section division be used for all tasks. The construction order can be changed at any time; all the dependencies of schedule tasks are preserved and the results of the changed order can be seen graphically at the optimised planning.

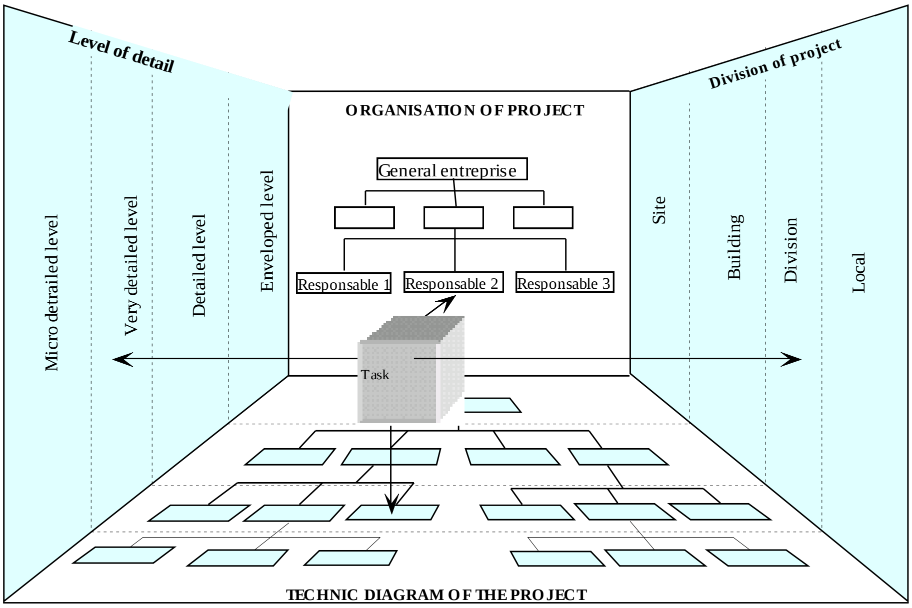


Figure 2: Project technical diagram

The sequence of activities is defined by precedence links between the activities in each unit, as established in the typical unit logic network (Figure 3). Thus, the identical units of a project are grouped together according to the logic constraints of superposed floors. For example, the second storey cannot be constructed before the first one and this one must be constructed after the ground storey.

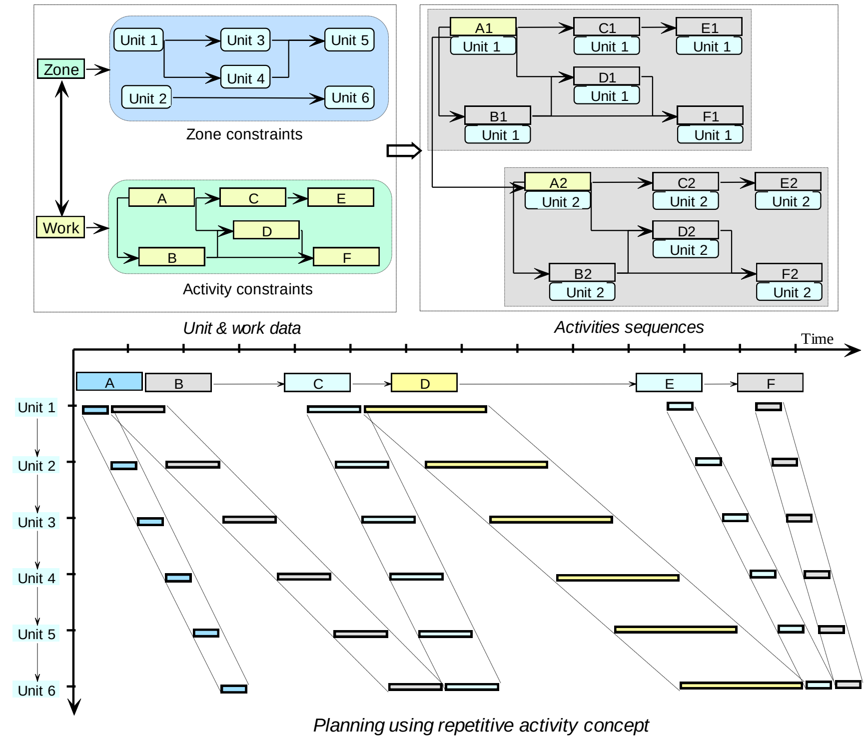


Figure 3: Zone and work constraints and their planning

On the other hand, a resource is used by several activities. In order to maintain the work team's continuity constraint, repetitive units must be scheduled to enable timely movements of teams from one unit to the next, avoiding work team delay time. Based on such constraints, an activity cannot be scheduled until all its preceding activities have not been scheduled. The constraints are a result of many factors, such as technological dependencies between activities, weather constraints, imposed dates, safety constraints, and so on. An example of such a constraint is a technological constraint, which needs the completion of columns and beams of a storey before finishing the slab for this storey. The model consists of a combination of activities, units and teamwork continuity constraints. The concept is presented in Figure 3. The sequence of activities can be obtained after checking all constraints (Figure 4).

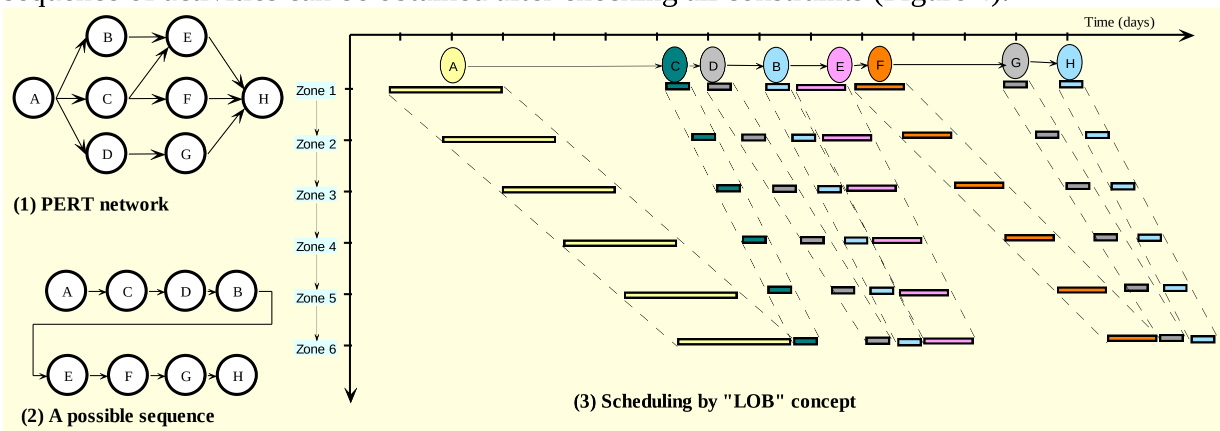


Figure 4: Repetitive activities concept

Concepts

A team work must move from one unit to another without dead time or interruption in order to avoid several arrivals to the building site. It supposes a simple adaptation of workflow rate. For two repetitive activities A and B presented in Figure 5-a, activity B is interrupted several times because it has a lower workflow rate than activity A. In order to avoid these dead time, individual activities of B should be delayed (Figure 3-b). The activity duration is given by Eq. (1):

$$d_A = \frac{Q_{Ar}}{U_A \times QR_{Ar}} \qquad (1)$$

d_A : duration of individual activity A; Q_{Ar} : quantity of activity A realized with resource r ; QR_{Ar} : quantity of resource needed for activity A; U_A : unit rate of activity A. The workflow rate for an activity is given by the Eq. (2):

$$C_A = \frac{d_A}{q_A} \qquad (2)$$

C_A : workflow rate of the activity A; q_A : number of teams for activity A.

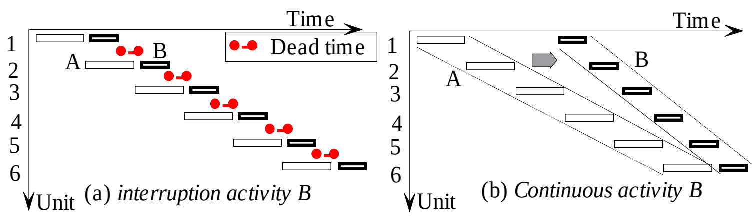


Figure 5: Teamwork continuity

The workflow rate can be used in order to adjust the duration of each activity. In fact, it depends on the number of teams or resource needed for an activity. These resources determine the activity duration. For example, increasing the team's number of an activity reduces its duration, which is given by Eq. (3):

$$D_A = |j - i| \times C_A + d_A \qquad (3)$$

D_A : repetitive activity duration, i : start unit of A; j : finish unit of A, (Figure 6-a). Manipulating the workflow rate minimizes or cancels a time lag between two activities (Figure 6,b,c,d).

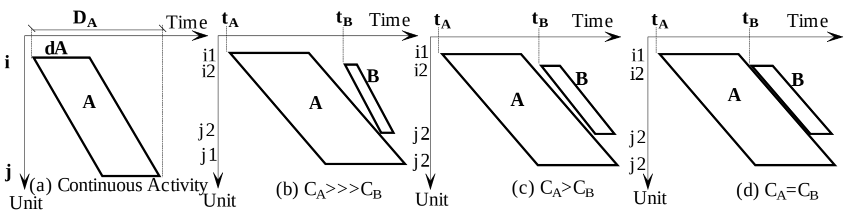


Figure 6: Repetitive activity and its workflow rate

Preplanning repetitive activities method/Control tools of repetitive activities constraints

Succession and imposed constraints should be satisfied in order to make a correct planning. A succession constraint is related with the sequence of activities and availability of teams. An imposed constraint is a constraint between two particular individual activities in different units. As a result, the constraints are checked in 5 cases in this model. They are the heart of our model.

- **Validation of succession constraints**

The satisfaction of a succession constraint for an activity consist in checking its starting date that must be equal or greater than completion dates of all preceding activities on every typical units. But, the validation of the activity doesn't need, in fact, to be made for every unit. The validation will be done for a particular unit, named validation point or critical unit. In fact, the validation of the constraint for this unit will insure the constraint validation for all other units.

The validation point is determined by the two activities to be checked, their continuity, their duration, their workflow direction and the units where they have to be done. The validation of succession constraints will be done through five cases. The cases from one to four present the determination of validation point when the two activities are repetitive. On the other hand, the fifth case is used when one at least of the two activities is individual.

In order to calculate the next activity starting date, the constraint between two successive activities A and B is given by the following Eq. (4) to be checked:

$$t_B - t_A \geq \delta_{AB} \quad (4)$$

t_A : Starting date of activity A; t_B : Starting date of activity B; δ_{AB} : Minimum time lag between the two starting dates of A and B.

Calculation of δ_{AB} : supposing that repetitive activities A and B are situated as presented in Figure 7. K is a unity of the two activities.

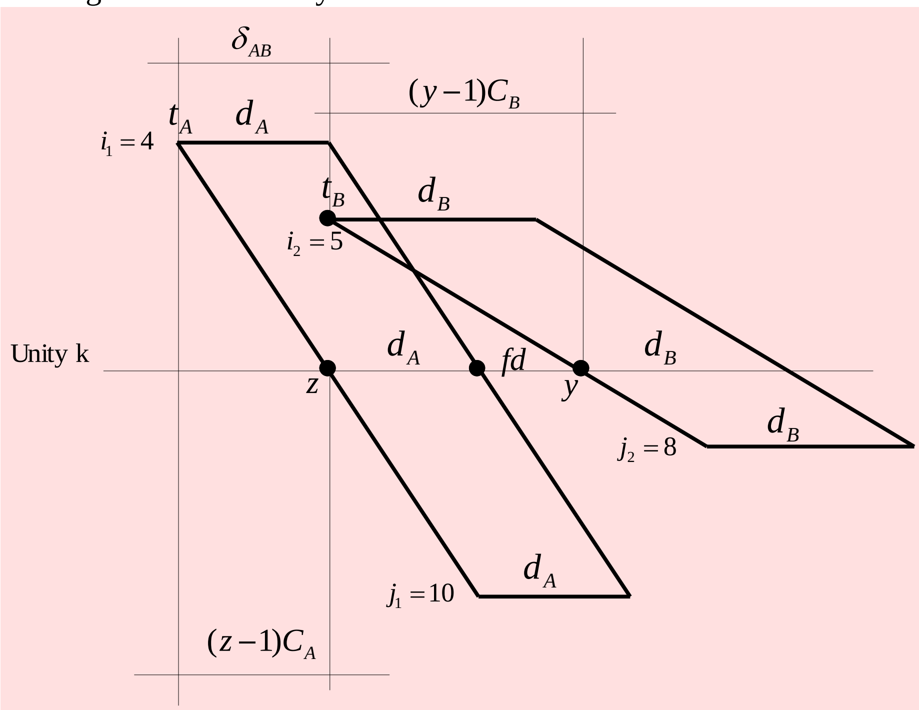


Figure 7 : Calculate of δ_{AB}

$k \in [i_1, j_1]$ and $k \in [i_2, j_2]$, $z = (k - i_1 + 1)$ and $y = (k - i_2 + 1)$

z and y represent the number of times where the individual tasks A and B were done to arrive to the unity K.

Figure 7 presents clearly that the beginning time of B can be determined by the following equation: $t_B - t_A \geq (z-1)C_A + d_A + fd - (y-1)C_B$

where *fd* is the constraint finish-start to be respected between the individual tasks A and B. This equation is applied to an unspecified unity that does not insure the succession constraint of tasks A and B in all the unities. It is thus necessary to apply this general equation to a particular unity, called critical unity which insures that the same constraint will be automatically satisfied in all the other unities. In the case of our example in Figure 7, it is clear that the critical unity is the first unity of the task B. The equation is thus applied with following values:

$k = i_2$ thus $\begin{matrix}(y-1)=0 \\ (z-1)=(i_2-i_1)\end{matrix}$ and $\delta_{AB} = (i_2 - i_1)C_A + d_A + fd$

Table I summaries all the cases for validation the succession constraints.

		Activity A		
	Activity type	Individual	Upward	Downward
Activity B	Individual	Case 5-5	Case 5-3	Case 5-4
	Upward	Case 5-1	Case 1	Case 3
	Downward	Case 5-2	Case 2	Case 4

- **Validation of imposed constraints**

The model can also validate imposed constraints. For each activity, the validation point is located according to the constraint that links two units together and is checked in result. This constraint represents the delay between two individual activities; four kinds can be defined according to the points of beginning and arrival of the constraint: finish-finish (ff), finish-start (fs), start-finish (sf) and start-start (ss).

The constraint joins the unit z in activity A with the unit y of activity B. The equations needed in order to validate the constraint and therefore the starting date of the following activity B is the following:

Finish-start (fs): $t_B - t_A \geq (z-1)C_A + d_A + fs - (y-1)C_B$ (22)

Start-finish (sf): $t_B - t_A \geq (z-1)C_A + sf - (y-1)C_B - d_B$ (23)

Start-start (ss): $t_B - t_A \geq (z-1)C_A + ss - (y-1)C_B$ (24)

Finish-finish (ff): $t_B - t_A \geq (z-1)C_A + d_A + ff - (y-1)C_B - d_B$ (25)

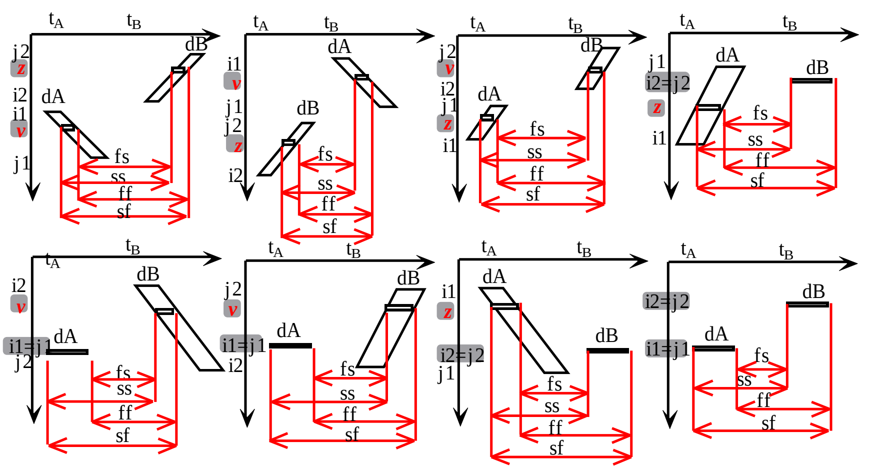


Figure 8: The imposing constraint between the activities

These equations are used to validate imposed constraints whatever the types of the two activities are $z=1$ or $y=1$ when the starting activity unit is one and $z = |j_1 - i_1| + 1$ or

$y = |j_2 - i_2| + 1$ when it is the last one. Figure 8 presents the different types of activities connected with the different imposed constraints.

Scheduling calculation

Multi unit projects can be scheduled using commonly accepted CPM technique, but a continuous utilisation of resources cannot be assured when these CPM networks are used. This shortcoming is best illustrated by an example. Figure 2 is CPM network repeated for a project consisting of six repeating units of works. The linking activities represent works that should be done from unit to unit using technical previous constraints in the network; for example; activities B1, C1 and D1 cannot be started until activity A1 is completed. The similar activities are mentioned in letters and the units in numbers. B1 means that the activity B is done in the first unit; A2 is the activity A in the second unit etc. The similar activities in different units are linked by resource availability constraints; for example activity A2 cannot begin until the workteam of activity A1 has finished.

The CPM network for one unit is used to establish the sequence of activities. This order is used to calculate the project duration. The activity sequence of (A-B-C-D-E-F) validates the constraints and the project duration can be obtained by the Eq. (26 & 27):

$$DD_j = \underset{i \in A_i}{Max}\left(DD_i + \delta_{i,j}\right) \quad (26)$$

$$DF_j = \underset{i \in A_i}{Max}\left(DD_j + D_j - 1\right) \quad (27)$$

DD_j : Starting date of activity j; DD_i : Starting date of activity i; DF_j : Finish date of activity j, $\delta_{i,j}$: Constraint between activities (i and j) that can concluded according to validation case. A_i : Ancestor activities of activity i.

Equations (26 and 27) allow obtaining the total duration by calculating the finish date of the last activity in the project. Activities sequence and possible scheduling planning are obtained randomly by possible solution of CPM network and identified in a database or entered by user as special constraints.

Application case

The realization of a repetitive task requires the repetition of the same process for each task in each unity of place throughout the project. For example, the realization of a bathroom is repeated of a unity to another to keep the continuity of the teams. To keep the continuity of the work of the teams in these processes, the unities of place must be put in a precise order to avoid the dead-time of the teams after finishing each task. The utilisation of such continuity benefits greatly the team work and minimizes the dead-time for each team at all repetitive activities tasks.

Continuity of teamwork could involve longer project duration because of the rigidity of resources utilisations. Some cases of scheduling have presented that the ignorance of the constraint of continuity of teams, by stopping each team when there is a need, in certain cases, can reduce the total duration and the indirect cost of the project which describes the indirect expenses of the building site like the telephone factor for example. Whereas the waiting of teams can also decrease the total duration of the project, they can increase the direct cost. A. D. Russel [18] criticized the application of the continuity of teams for all the activities of the project and he proposed that it is recommended for certain important tasks. The traditional methods of repetitive activities scheduling had been criticized for their weaknesses of respecting the use continues resources [19], [15] and [18].

The method that we proposed for managing repetitive activities is based on the validation of two types of constraints. These constraints are the succession constraints and teamwork continuity constraints. The succession constraint is identified by the relations between the activities according to the method of task realization. The second constraint is the availability of the teams which depends on the number of teams affected for the realization of a task. For example, using only one team for painting 6 floors, obliges the team to move towards the following floor upon finishing the current floor.

Thus, several factors must be taken into account in the planning phase: the type of repetitive activities, the number of the teams, the interruption of the teams, the availability of the teams, the sequence of realization of work on the unities and the number of workers in one team. The combination between these factors creates a great number of possible solutions for planning.

In order to present the impact at the planning, we study the example presented in figures 9 to 16. Figure 9 and Table 1 presents the initial case of scheduling of six continuous upwards activities. The duration of this project is 30 days. We will study the effects of the change of some factors.

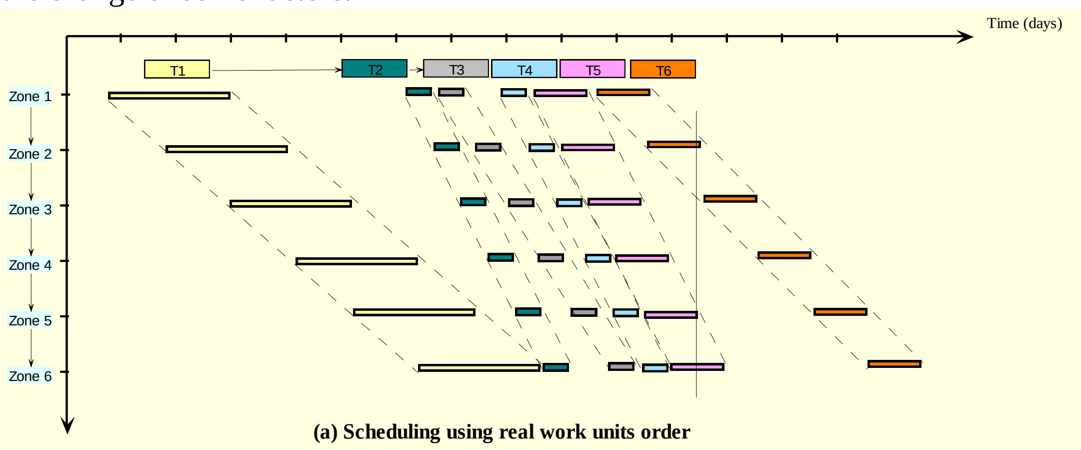
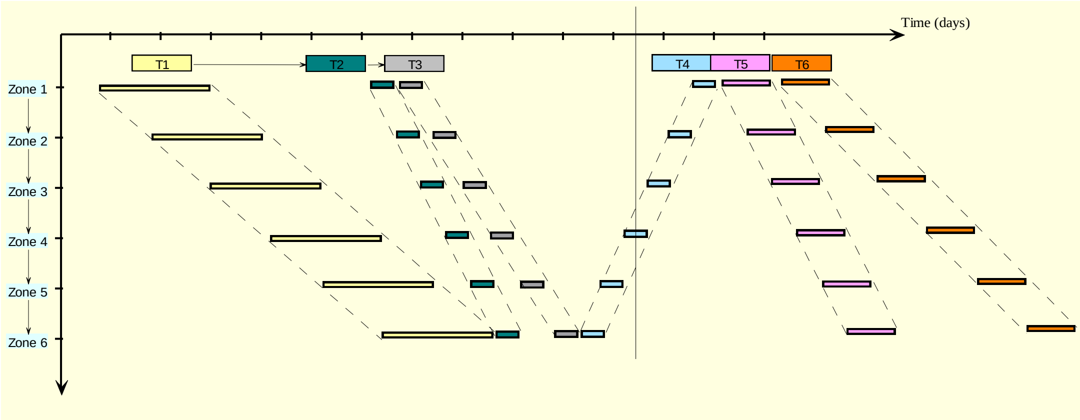


Figure 9 : The initial case of scheduling of six continuous upwards activities

Table 1 : Initial case of scheduling of six continuous upwards activities.

Activity	Duration	Teams number	Start unity	Finish unity	Workflow rate	Start time	Finish time
T1	3	2	1	6	1,5	1	11,5
T2	1	1	1	6	1	7	13
T3	2	1	1	6	2	8	20
T4	1	1	1	6	1	15	21
T5	2	2	1	6	1	16	23
T6	2	1	1	6	2	18	30

Figure 10 and Table 2 show the result for changing the type of the fourth activity by a downward one. The duration of project increases (10) days to be (40) days.



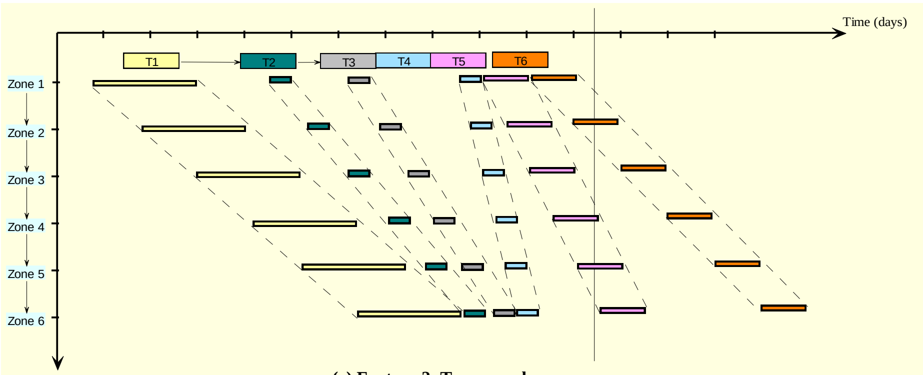
(b) Facteur 1: Type of activity

Figure 10 : Factor 1: Type of Activity

Table 2 : Factor 1 type of Activity

Activity	Duration	Teams number	Start unity	Finish unity	Workflow rate	Start time	Finish time
T1	3	2	1	6	1,5	1	11,5
T2	1	1	1	6	1	7	13
T3	2	1	1	6	2	8	20
T4	1	1	6	1	1	20	26
T5	2	2	1	6	1	26	33
T6	2	1	1	6	2	28	40

In other hand, adding a team on the realization of activity number 4, the duration of project increased (3) days and it becomes (33) days (see Figure 11 and Table 3).



(c) Facteur 2: Team numbers

Figure 12: Factor 2: Team Numbers

Table 3 : Factor 2 team numbers

Activity	Duration	Teams number	Start unity	Finish unity	Workflow rate	Start time	Finish time
T1	3	2	1	6	1,5	1	11,5
T2	1	1	1	6	1	7	13
T3	2	1	1	6	2	8	20
T4	1	2	1	6	0,5	18	21,5
T5	2	2	1	6	1	19	26
T6	2	1	1	6	2	21	33

The impact of interruption of the teams is illustrated in Figure 13 and Table 4 which gives (30) days for the total duration of the project. The interruption of activity number 2 does not have any influence at the total duration of project but the teams are stopped in each zone and there is a waste of time in which they don't work.

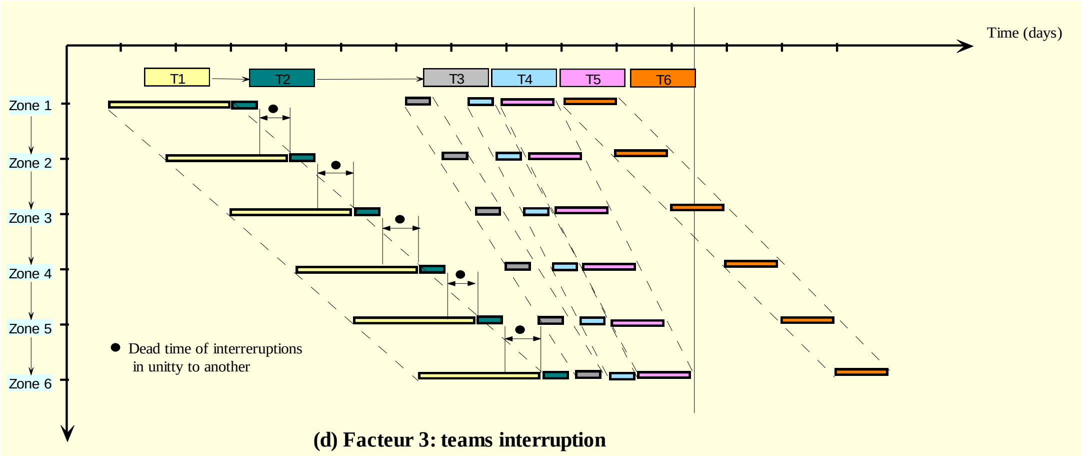


Figure 13: factor 3: teams interruption

Table 4 : Factor 3 teams interruption

Activity	Duration	Teams number	Start unity	Finish unity	Workflow rate	Start time	Finish time
T1	3	2	1	6	1,5	1	11,5
T2	1	1	1	6	1	7	13
T3	2	1	1	6	2	8	20
T4	1	1	1	6	1	15	21
T5	2	2	1	6	1	16	23
T6	2	1	1	6	2	18	30

The fourth factor is the team's availability at the building site that requires shifting activity 2 if it shares the same resources with activity 1. It is necessary that the team of task 1 finishes and passes to task 2 (Figure 14 and Table 5).

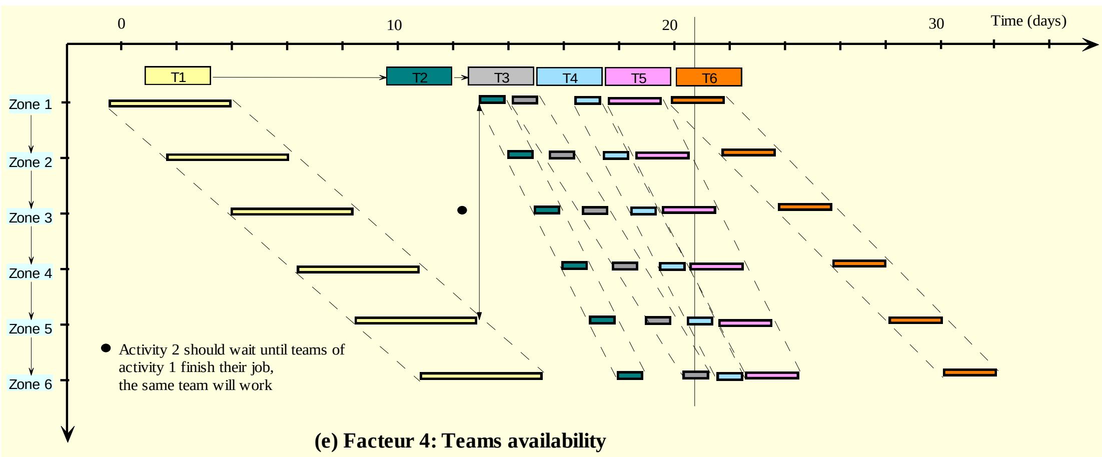


Figure 14: factor 4: Teams availability

Table 5 : Factor 4 Teams availability

Activity	Duration	Teams number	Start unity	Finish unity	Workflow rate	Start time	Finish time
T1	3	2	1	6	1,5	1	11,5
T2	1	1	1	6	1	12	18
T3	2	1	1	6	2	13	25
T4	1	1	1	6	1	20	26
T5	2	2	1	6	1	21	28
T6	2	1	1	6	2	22	35

The fifth factor is the order of tasks. Changing the sequence of activities can influence by decreasing the project duration (Figure 15 and Figure 6). This is what happens in our example by changing the position of task 6.

.

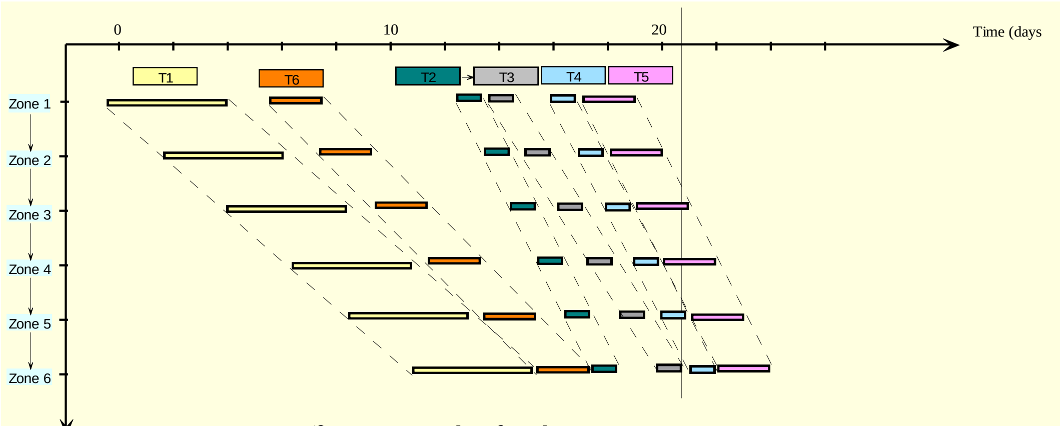


Figure 15: Factor 5: ordre of works

Table 6 : Factor 5 order of works

Activity	Duration	Teams number	Start unity	Finish unity	Workflow rate	Start time	Finish time
T1	3	2	1	6	1,5	1	11,5
T6	2	1	1	6	2	4	16
T2	1	1	1	6	1	11	17
T3	2	1	1	6	2	12	24
T4	1	1	6	1	1	19	25
T5	2	2	1	6	1	20	27

The last factor relates to the individual task itself. In fact, the duration of a task does not stay the same by adding a worker or several workers to its teams. It decreases the individual duration of the task, which has important consequences at the project duration. Figure 16 illustrates the addition of one worker at each team of task 1. The individual duration of the task becomes 2 days, which increases the project duration to (26) days (Tableau 7). In this case, the addition labour has an important influence over the duration of task as well as the project duration. Additional workers may in certain cases increase the individual activity duration because of the high density of labour in the same unity.

The combination between these factors influences the project duration and sometimes produces very good results.

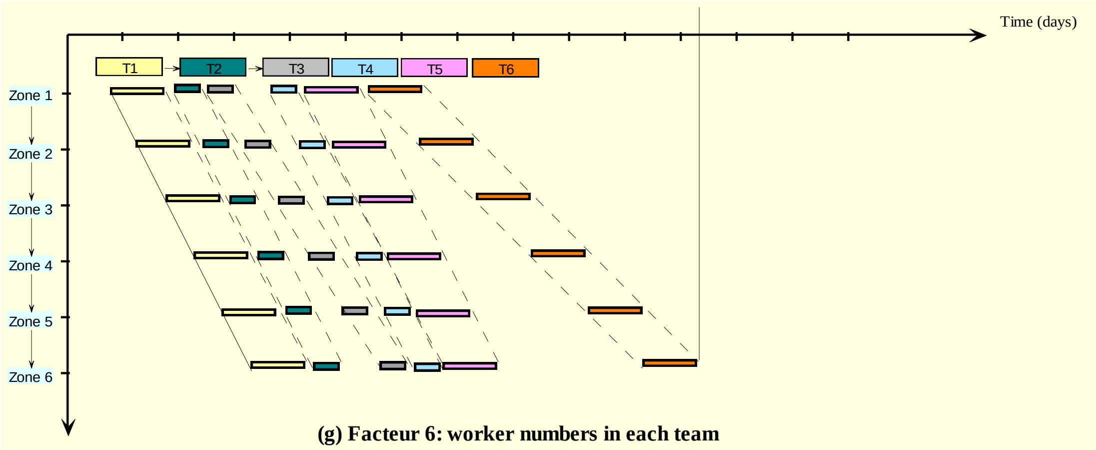


Figure 16: Factor 6: worker numbers in each team

Table 7 : Factor 6 worker numbers in each team

Activity	Duration	Teams number	Start unity	Finish unity	Workflow rate	Start time	Finish time
T1	2	2	1	6	1,5	1	8
T2	1	1	1	6	1	3	9
T3	2	1	1	6	2	4	16
T4	1	1	1	6	1	11	17
T5	2	2	1	6	1	12	19
T6	2	1	1	6	2	14	26

The existing methods suggested to schedule the repetitive tasks do not take into account all these factors. They cannot find an optimised solution. The algorithm that we propose, can establish the optimised scheduling of the start and finish dates of each activity in a practical and simple way taking into account all these factors.

Illustrative example of the algorithm of management of resources teams of repetitive tasks

We explain the algorithm of continuity constraint validation for the teams by an example. The two successive tasks need the same teams. The validation requires validating the three constraints of succession constraint, availability of teams and continuity of work (Figure 17).

The first step of our algorithm is based on scheduling the activity B. In fact, this scheduling is done in respecting the continuity of teams that realising reach individual activity of B in each unity. The second constraint to be taken in account is that the individual activities of B will start directly after finishing activity A in each unity. To respect this condition we can after scheduling activities B directly that the continuity of work team is not respected (Figure IV. 17-a). To keep this flexible movement of the teams, the activity B2, B3, B4, B5 and B6 should be shifted but the succession constraint is no more respected (Figure IV. 17-b).

To avoid theses constraints, the first stage of our algorithm places the individual activities of B in repetitive ones, so that they become like a block of activities. The application of our system of validation of constraints becomes real. So, this procedure

guarantees the validation of the constraints of continuity of the team and succession (Figure IV. 17-c).
The second step is to validate the constraint of team availability. The two activities (A and B) need the same teams. Thus, B1 is shifted so that it starts after the end of A6. In order to keep the constraint of work team continuity, all B activities should be shifted (Figure IV. 17-d).

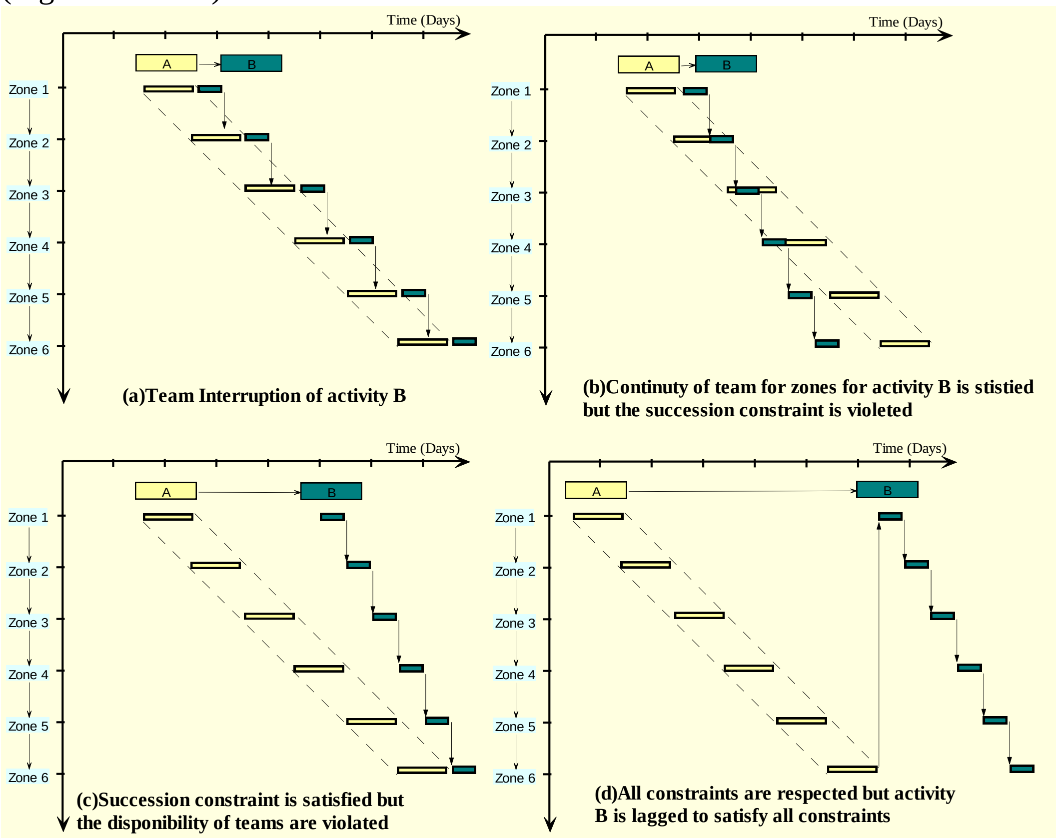


Figure 17: Etapes d'algorithmes de validation de différentes contraintes

This algorithm is applied at all constraints validation, continuity and availability of the teams.

Model of project cost:

Most contractors have a director who is responsible for the estimation section and this fact indicates the importance of estimation function based on resources utilisation during the project execution. Our model takes into account the project direct costs, which are identified in the planning preparation. Several project estimations can be obtained according to the different realization options, duration and planning management. The global cost is evaluated from the direct costs of daily real resource quantities. It consists of material cost, plant cost, labour cost (taxes, studies, insurance, etc) and penalty costs. The global estimation is assured by the Eq. (28), which avoids errors occurred by unproductive resources times [14].

$$P = p^{ma} + p^{rr} + p^{mc} + FC + FS + Penalty \qquad (28)$$

p^{ma}: Cost of material (consumable resources); p^{rr}: Cost of renewable resources; p^{mc}: Plant cost of work site; FC: Work site cost; FS: Special costs; Penalty: Penalty costs of overtaking project duration.

GA background

GAs have been popularly used in many areas, such as constrained or unconstrained optimisation, scheduling and sequencing, transportation, reliability optimisation, artificial intelligence, and many others fields [8] [2] [3] [13]. Some researches using GAs have been done in the optimisation of construction scheduling [5], [7], [9] and [11]. GAs are stochastic search techniques based on the mechanism of natural selection and natural genetics. Genetic algorithms are working as shown in (Figure 18). In GAs, potential solutions to a problem are represented as a population of chromosomes and each chromosome stands for a possible solution at hand. The chromosomes evolve through successive generations. Offspring chromosomes are created by merging two parent chromosomes using a crossover operator, or modifying a chromosome using a mutation operator. During each generation, the chromosomes are evaluated on their performances with respect to the fitness functions (i.e., objective functions). Chromosomes that are fitter have higher survival probabilities and the reproduction of chromosomes is lead through a specific selection rule. After several generations, better and better chromosomes appear. At the end, after a given number of generations, the final chromosomes hopefully represent the optimal or near-optimal solutions of the problem (Figure 18).

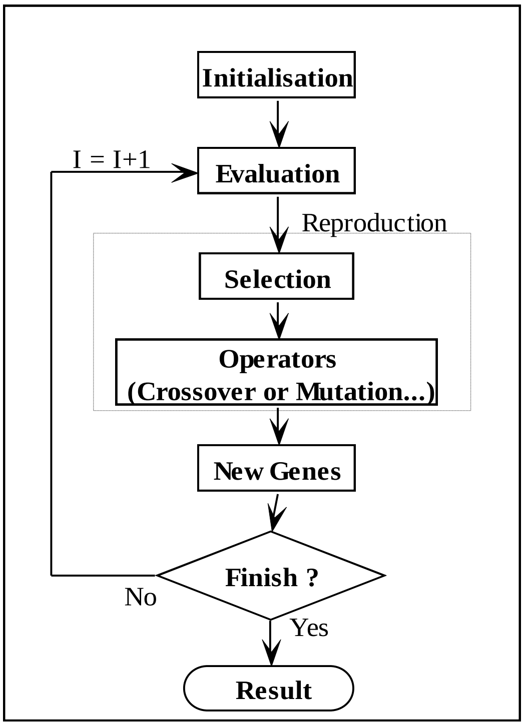


Figure 18: Operation of genetic algorithms

Optimisation by genetic algorithms:

Different possible activities durations depend on different realization options and different team organisation in project site, which lead to continuous or intermittent activity. Possible realization choices (continuous or intermittent activities) give different project durations and consequently different project costs.

The basic work in repetitive activities scheduling is to decide the order in which the activities should be performed. A schedule network, consider the graph given at Figure 19 as an example, gives the technological constraints that must be respected in each unit. For instance, in Figure 19, activity B must be realized after activity A in each unit. It doesn't mean that activity A must completely precedes activity B. On the basis of this network, different sequences of activities can be built. Figure 19 shows one of these sequences. For a real problem, the number of possible sequences is often very great. The interest of using GAs to solve repetitive activities scheduling problems is to find easily the best sequences related to a declared objective (duration or cost of the project for instance) to be optimised.

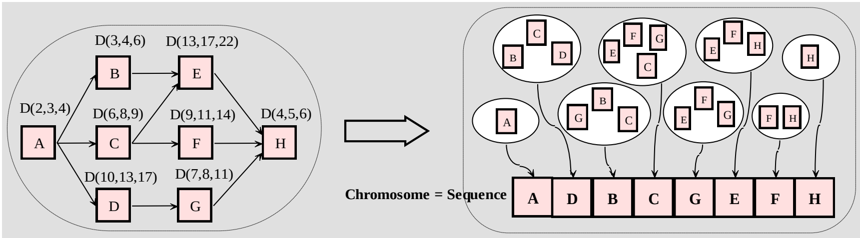


Figure 19: Encoding the sequence of chromosome

Each possible sequence of activities is in fact a chromosome; the number of genes of this chromosome is represented by the number of activities to be scheduled. This sequence or chromosome is also used to solve the problem of allocation resources. Because resources are always limited, a conflict between activities can appear for resource allocation. It is solved through the order of the activities given in the sequence. In the sequence of figure 19, activity B is in a lower position than activity D. It means that activity D has a higher priority than activity B for resources, i.e. the common resources are firstly affected to activity D that is scheduled before activity B in each unit.

- **Encoding the chromosome**

Each chromosome represents a solution of a possible planning. Each gene carries out an activity of the project. Figure 21 presents a chromosome composed of eight genes, which means eight activities. The order in a chromosome takes into account CPM network constraints between activities. Each gene represents an activity and its value is possible activity duration according to realisation options. For example in Figure 21, the possible durations for activity C are 9, 8 or 6 days. We have distinguished three types of chromosomes: (1) activities order chromosome (Figure 21), (2) project cost and duration chromosome (Figure 21.a) and (3) resource smoothing chromosome (Figure 21.b).

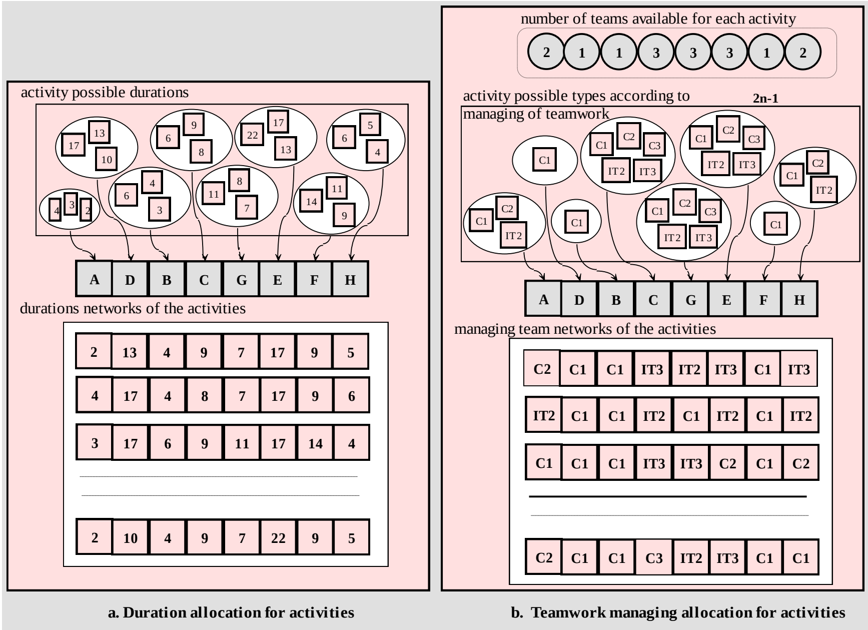


Figure 20: Encoding activity duration/cost and teams managing

As said before, the sequence of genes defines the priority order of resources allocation. An activity located in the chromosome left edge presents the highest priority of execution order. The degree of priority of an activity realisation decreases progressively from left to right. Each sequence is generated randomly using the operators of genetic algorithms reproduction respecting every constraint.

Activity “A” is composed of several possible durations in which we find three values (4, 3, 2) (Figure 20.a). Each value represents a realisation option. For activity B possible durations are (6, 4, 3). A different combination of genes defines a possible planning and consequently generates different project durations. For each construction activity, the present model is designed to consider all relevant decision variables that may have an impact on project time and/or cost. This includes construction methods, which indicate the availability of different types of materials and/or techniques that can be used. It includes also teams organisation, which represents feasible sizes and configurations for construction teamwork. In order to control the complexity of the optimised model, our prototype combines the two major decision variables into single decision variable called resource utilization. For example, a small-size project that includes 20 activities and 5 possible resources utilization options for each activity creates a search space of approximately 95 trillion (i.e., 520) possible solutions. Thus, the present model is designed to help planners in the challenging tasks that require a large solution space using genetic algorithms technique in order to identify optimal resources utilization plans that achieve project objectives (Figure 20.b).

In order to apply genetic algorithms to solve the problem of making repetitive activities planning, another chromosome is needed to smooth the solution in which genes represent the number of teams and the activity type. Figure 14 shows this chromosome. Each gene has two initials: a letter that identifies the activity types, C for continuous and IT for intermittent and a number that stands for the number of teams. For example, “C1” for the activity “A” means that activity A is continuous with one team, “IT2” that the activity is intermittent and divided in two teams (Figure 21).

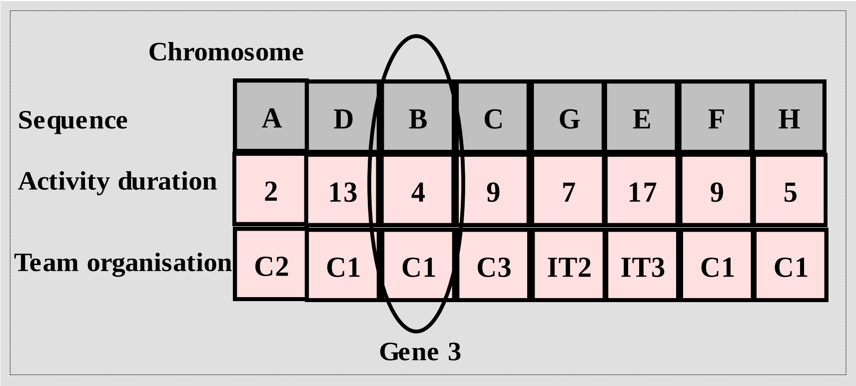


Figure 21: Encoding the chromosome according to sequence, duration and type of activities

- **Evaluation function:**

The evaluation of each chromosome is based on an adaptation function presented previously in order to choose in continuation the chromosomes by a selection rule: here the "N/2-elitism" rule. This method neglects the weakest chromosomes in the present generation and considers only the half strongest chromosomes, related to the adaptation score value, to create the next generation. The different chromosomes or solutions are thus classified according to their adaptation score values.

The evaluation can be done for the two most important criteria that are used in practice: time and cost for project realization.

In order to select the strongest individuals as said before, we calculate for chromosome its adaptation score (SA) obtained by the weighting method between duration and cost of the project. The most powerful individual is characterized by the most important adaptation score value.

SA is calculated as following. First, a standardized vector is calculated for duration (Eq. 29) and for cost (Eq.30):

$$\mathrm{VD}_s = \frac{1/\mathrm{DT}_s}{\sum_s 1/\mathrm{DT}_s} \qquad (29)$$

VD_s : Standardized vector of activity durations of the chromosome s

DT_s: Total duration of the project for chromosome s

$$\mathrm{VC}_s = \frac{1/\mathrm{CT}_s}{\sum_s 1/\mathrm{CT}_s} \qquad (30)$$

VC_s: Standardized vector of the costs of the chromosome s

CT_s: Total cost of the project for chromosome s

Each chromosome is evaluated by its adaptation score, which is given by (Eq. 31)

$$\mathrm{SA}_s = \alpha.VD_s + (1-\alpha).VC_S \qquad \alpha \in [0,1] \qquad (31)$$

It is so possible to balance between project duration and project cost as wanted.

$\alpha = 1 \quad \Rightarrow SA_s = VD_s$: Optimisation with project duration only

$\alpha = 0 \quad \Rightarrow SA_s = VC_s$: Optimisation with project cost only

$\alpha = 0.5 \qquad \Rightarrow SA_s = \frac{1}{2}[VD_s + VC_s]$: Optimisation with the same weight for cost and duration.

The most powerful individual is characterized by the most important adaptation score value, which means that this solution is more adopted than the others. We can thus arrange the solutions according to their adaptation score values.

So, our evaluation function is based on a trade-off between project duration and cost. The best planning schedule, which is based on activities options and therefore activities durations and teamworks, must give an optimal or near-optimal project cost and duration. Each new generation created with crossover and mutation operators produces new chains of possible planning solutions that should be evaluated in cost and time SA = f (time, cost). Genetic algorithms are used to obtain the finest possible solution. It works with the following methodology:

Generation of the initial population:

To initialise the algorithm, a population of chromosomes with a fixed size is generated randomly (Figure 16). It is uniformly shared on the search space. Each chromosome is then evaluated according to the fitness objective function SA.

Selection of the chromosomes:

The selection operation uses the results chromosomes of generation (t) to construct the (t+1)th generation. The use of the superior half of the previous generation (N/2 elitism) allows obtaining a more adaptative generation with regard to the evaluation function. The selection process is designed to maintain systematically the finest chromosomes of the current population into the following generation (Figure 22).

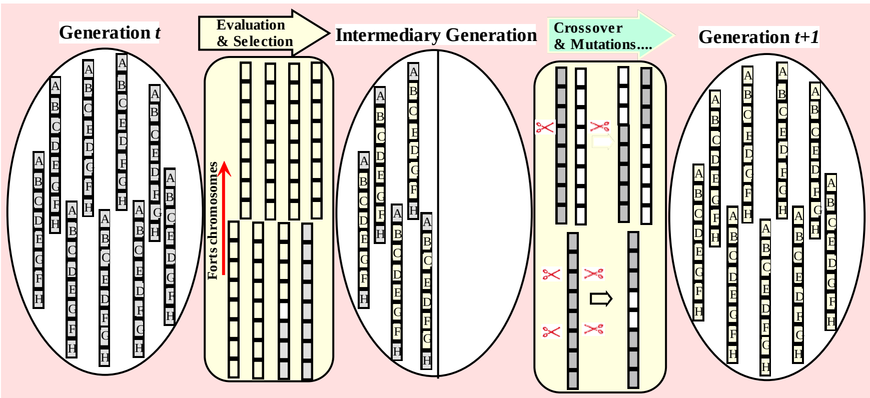


Figure 22: N/2- elitism selection

Crossover and mutation operators for scheduling activities

The chromosome construction starts by selecting activities without predecessors. If there are several activities of first order, one of them is chosen randomly. Selected activity takes place in the first left place of the chromosome and is deleted from the activity list. If the chromosome is not completed, another activity is chosen from remainder possible activities and is put in the next place. For instance in Figure 23, the possible successors activities of activity "D" are put in the roulette (B, C and G) and one of them will be chosen, i.e. activity B. This algorithm is repeated to put all the activities in the chromosome (Figure 23).

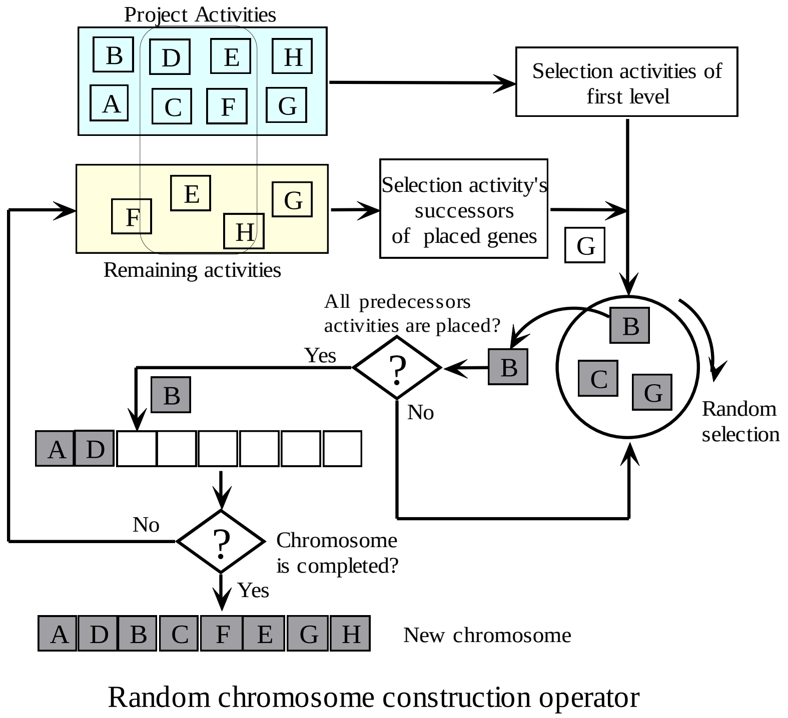


Figure 23: Random chromosome construction operator

The mutation operator is used to make a new chromosome (Figure 23). It changes genes respecting succession constraints. Supposing that a mutation will give an individual A-B-C-D-E-F-G-H. Two cuts are randomly chosen to determinate genes between them. In our example, the two cuts are placed on sites 2 and 6. The genes C-D-E-F are taken off from the original chromosome. One of these genes is randomly chosen and put in the chromosome according to his predecessors respecting constraints between activities. If not, another gene is chosen in the temporary list, until all activities are placed.

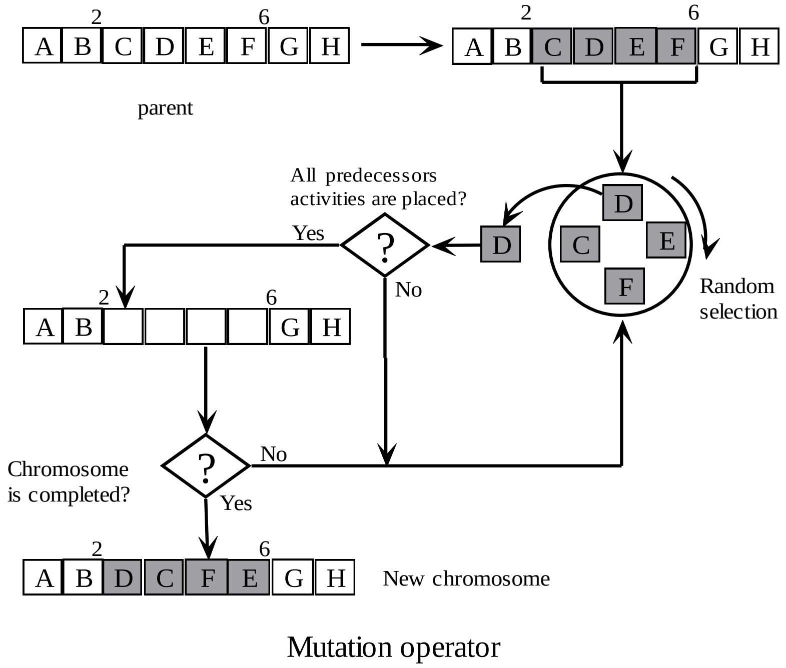


Figure 24: Random mutation operator

One of the main problems with genetic algorithms for scheduling problems is to identify the operators able to product a new feasible chromosome. The crossover operator adopted in our work is related with the UM3 operator adopted by (Leu, Yang) [12]. Some differences have been introduced. It means that two parents are picked up randomly and the cutting position is also randomly chosen. Then, the two parts of different parents are pasted together. The two children results of this operator must be tested and validated or refused.

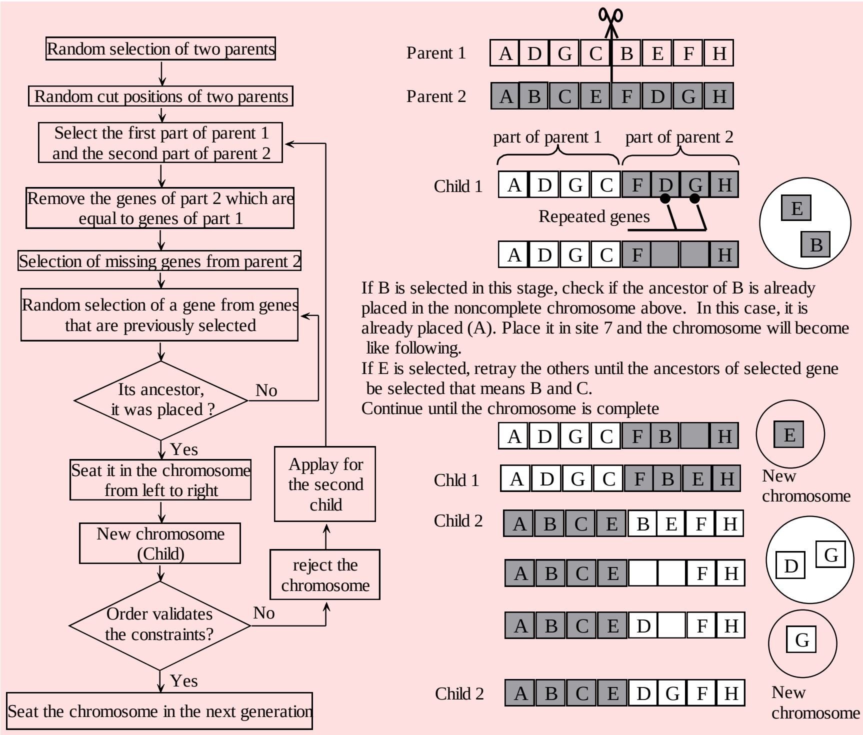


Figure 25: Crossover operator

The two children can be refused for two main reasons:

- If the activities don't validate the constraints, or
- If there are repetitions for some activities in the same chromosome.

If a child chromosome is not satisfactory, it is not directly refused. In fact, we try first to suppress repetitions and see, in a second time, if the technological constraints are satisfied. The correction operator is based on a suppression of the repeated activities and an introduction of the activities that don't appear in the chromosome. One of these missing activities is chosen and checked if all its predecessors have been chosen in the chromosome. If not, another activity will be chosen. This process is continued until every activity falls at a specific place in the chromosome. The chromosome is then tested for succession constraints. If it validates these constraints, it will be a part of the next generation. If not, this chromosome is eliminated and thus two cuts are made in order to gain some time in the population generation. Figure 25 presents our crossover operator.

Conclusion and perspectives

The approach proposed in this paper allows the construction of a set of solutions for project planning with repetitive and non-repetitive activities. Activities relations, multiple teamwork strategies and time-cost estimation are considered in a specific model built for non-repetitive activities scheduling. An optimisation formulation is presented for tasks scheduling with the objective to minimize total construction cost or time, or to find a trade-off between time and cost.

This problem of repetitive activities scheduling is, indeed, a combinative problem, which can be solved by genetic algorithms (GAs) with great efficiency.

A possible schedule can be easily described with three associated chromosomes, whose genes are able to represent the order of the activities in a sequence, the possible durations for these activities and different ways of organization for the work teams on the site.

The generation of an initial population respects the constraints between the activities. Each solution or chromosome obtained with the crossover operator is tested for constraints validation. The development of a new generation is based on the most powerful half ("N/2 elitism") of the precedent generation and assumed by two operators of crossing and mutation especially created for this problem. The change is made on several genes to ensure a good divergence of the different generations. Calculation time for optimisation increases according to the number of generations and according to the duration of complexity of constraints between the activities. It also increases significantly with the number of activities of the project. Nevertheless, large construction projects with a great number of activities can be treated.

The main results we begin to obtain on different construction projects are the following:

- it is possible to obtain significant profits on time with a good organisation of the activities and of the work teams; for instance, the notion of intermittent activity is very useful for an activity with a long duration placed between two shorter activities; it is also the case for a short activity between two longer tasks;
- a large scale solution can quickly be explored with GAs and so, original solutions can be detected; for instance, solutions with good continuity of the work of the different teams can be found.

Now, a large program of tests for real projects will soon begin with different French companies. They have two main objectives: to reduce total duration of the projects, to obtain a better productivity of the work teams.

Another perspective is to study the relation between time and cost. The GAs method allows generating different solutions evaluated with two criteria: the total cost and the global duration. It is possible to build a set of non-dominated solutions and to develop a multi-criteria analysis in order to find global rules of management. This will be a very interesting study for the future.

References

[1] D.B. Ashley, Simulation of repetitive unit construction, Journal of the Construction Division CO2 (1980) 185-194.

[2] D. Beasley, D.R. Bull, R.R. Martin, An Overview of Genetic Algorithms: Part 1, Research Topics, in: (Unversity Computing, 1993a).

[3] D. Beasley, D.R. Bull, R.R. Martin, An Overview of Genetic Algorithms: Part 2, Research Topics, in: (Unversity Computing, 1993b).

[4] G.E. Birrell, Construction planning: beyond the critical path, Journal of the Construction Division CO3 (1980) 389-407.
[5] W. Chan, D.K.H. Chua, G. Kannan, Construction Resource Scheduling With Genetic Algorithms, Journal of Construction Engineering and Management 122, N°2 (1996) 125-132.
[6] H.L. Fang, Genetic Algorithms in Timetabling and SchedulingUniversity of Edinburgh, 1994.
[7] C.W. Feng, L. Liu, S.A. Burns, Using genetic algorithms to solve construction time-cost tradeoff problems, Journal Construction Engeneering and Management, 11, N°3 (1997) 184-189.
[8] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, optimization and machine learning (1989) 432.
[9] T. Hegazy, Optimization of Resource Allocation and Leveling Using Genetic Algorithms, Journal of Construction Engineering and Management 125,N°3 (1999) 167-175.
[10] T. Hegazy, P. Fazio, M. O., BAL: An Algorithm for Scheduling and Control of Linear Projects, (1993).
[11] A. Istanbouli, J.-C. Mangin, Use of Genetic Algorithms to Optimise the Eriction Cycles on Building Sites, in: The 3rd International Conference on Decision Making in Urban and Civil Engineering (London, 2002).
[12] S.-S. Leu, C.-H. Yang, GA-Based Multicriteria Optimal Model for Construction Scheduling, Journal of Construction Engineering and Management 125, N°6 (1999) 420 - 427.
[13] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs (Springer-Verlag, Berlin, 1994).
[14] I.A.A. Putera, Génération et évaluation des activités de construction des bâtiments:Optimisation de la planification par les algorithmes génétiquesUniversité de Savoie, 2002.
[15] M.R. Rehab, RPM: repetitive project modelling, Journal of Construction Engineering and Management 2 (1990) 316± 330.
[16] M.J. Ricardo, F. Luiz, M. Heineck, Preplanning Method for Multi-Storey Building Construction using Line of Balance, in: 6th Annual Conference of the International Group for Lean Construction (Guarujá-Brazil, 1998).
[17] B.H. Robert, G.I. Photios, Scheduling Projects With Repeating Activities, Journal of Construction Engineering and Management 124 (1998) 269-278.
[18] A.D. Russell, W.C.M. Wong, New generation of planning structures, Journal of Construction Engineering and Management 2 (1993) 196± 214.
[19] S. Selinger, Construction planning for linear projects, Journal of the Construction Division CO2 (1980) 195± 205.
[20] W.Y. Thabet, Y.J. Beliveau, HVLS: Horizontal and Vertical Logic Scheduling for Multistory Projects, Journal Construction Engeneering and Management 120 (1994) 875-892.