LAYING SIEGE TO F//C^{max} FLOW-SHOP PROBLEM – IS 'RANDOM' THE LIKELY WINNER?

Ph.D. Zoltán A.Vattai
Budapest University of Technology and Economics, Faculty of Architecture
zvattai@ekt.bme.hu

Abstract

One of the most famous and challenging set of problems of Operations Research is the family of so called Flow-Shop Problems, where *n* pieces of products are to be scheduled for production using uniform technology on *m* pieces of machines for to achieve e.g. the shortest overall production time, or the minimum of waiting times, or some other target functions. Analogy in construction can be set when considering series of buildings or of construction elements (products) to be built in cooperation of series of (sub)contractors or of teams of resources (machines). Differing feature is that action of machines can be overlapped in time at processing the same product (building). The paper reports some interesting trials of finding solution for the problem, testing series of algorithms, such as Johnson's (1954) algorithm, variants of enumerations, pair-wise exchanges, and random sampling. Revealing conclusion of examinations has driven attention of researchers into direction of strict estimates on theoretical minimum of target values functioning potentially as criterion of optimum.

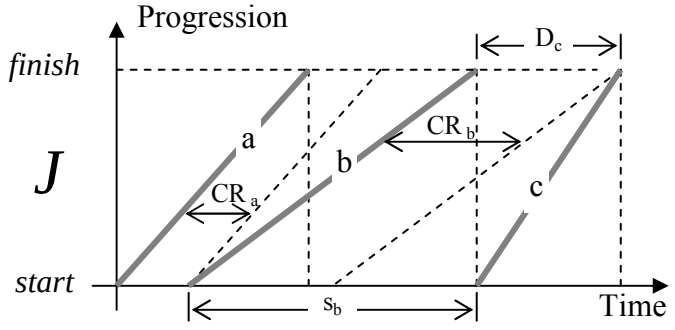
Keywords: operations research, flow-shop problem, construction modelling

PREFACE

For graphical representations of ideas to be discussed on following pages instead of using one-dimensional graphics of schedules (known as Gantt Charts), that is usual at discussing scheduling (mean: sequencing) problems, we indicate our thoughts on two-dimensional „time-route" diagrams known as „progression curves", „cyclograms" or as „linear schedules", as seen below. Horizontal axis of the diagrams is Time, while along vertical axis Progression can be read in any proper dimension such as m, m^2, m^3, ton, %, €, etc. that is a common characteristic of all processes (jobs) scheduled.

Figure 1.

Two-dimensional representation of timing processes (a,b,c) on project („workpiece") *J*



Processes are represented by individual lines (See: line „a", „b" and „c" above) slopes of which can be read as intensity of progression, while durations of processes (D_a, D_b, D_c) and timing of them (e.g. succession times between either at start: s_a, s_b or at finish: f_a, f_b) can be read as horizontal (time) views of their linear representations.

For succeeding processes with or without overlapping in time, minimum succession times (called „technology breaks", denoted as „CR_i" – read as „critical approach after process *i*") are set, defining expected minimums of non-overlapping periods.

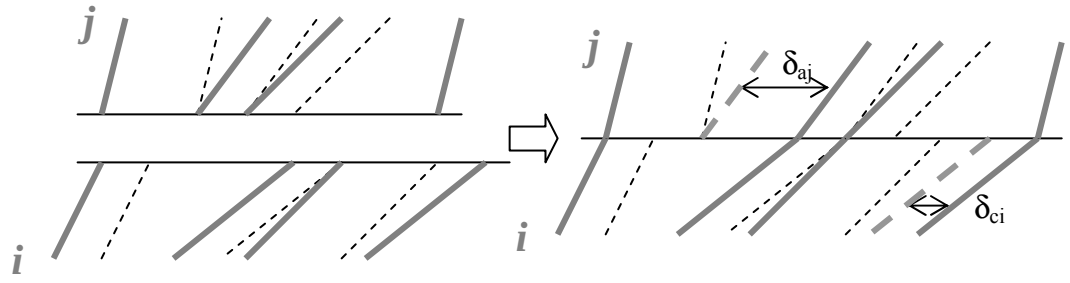
Ranges/values of critical approaches can be defined/selected tipically by considering technical-technological needs, such as needed time for hardening, cooling, drying, consolidating, etc. as constant values or can be predicted as functions of progression (processing times) due to needed manipulation area or safety zones or on-site storage capacities etc. regarding the manner of operation under consideration. Guiding relative timing of succeeding processes this way any relative time position, „lead" or „lag" time, can be set. Expectation of non-overlapping in time also can be considered as a special case of „overlapping", with succession time equal to minimum of durations of processes linked to each-other. (See process „b" and „c" in Figure 1. above – and CR_b between them. $CR_b=\min\{D_b, D_c\}$)

Minimum succession times (s_i, f_i) between succeeding processes (*i*, *i*+1) can simply be calculated using durations (D_i, D_{i+1}) and critical approach (CR_i) as input values:

$$s_i = \max\{CR_i,\ D_i+CR_i-D_{i+1}\};\quad f_i = \max\{CR_i,\ D_{i+1}+CR_i-D_i\}$$

In our context lines of processes in a schedule of a project (or of a workpiece) must not coincide or cross each-other, otherwise it would mean a partly differing (turning) technological order, that we exclude from our investigations. A project (workpiece) is complete, when the last process is completed. The schedule of a project in which all the succession times between neighbouring processes are at their minimum, set by „CR" values predicted, we do refer as „Own Schedule" or „most compact" schedule of the project/workpiece (as if it would be processed in its own). These most compact own-schedules of projects will be released by need (succession times between neighbouring processes can and will be increased if needed) when combining/linking more of them in a „Master Schedule". (See: Figure 2.)

Figure 2. Releasing Own Schedules of projects/workpieces (i, j) for linking them up in a Master Schedule with expectation of breakless performance of processes/machines. (Detail of Master Schedule is right to the arrow.)



Thus the problem of forming a Master Schedule and finding optimal sequence of projects for to achieve the aim preset is derived back to the problem of matching pairs of succesion vectors (succession times „$\underline{f}_i$" at finish on preceeding project/workpiece and succession times „$\underline{s}_j$" at start on succeeding project/workpiece).

INTRODUCTION

For testing and demonstrating effect of „Sequence" (of projects) on minimum overall execution time of a Master Schedule comprehending execution of numerous building projects („multi-project management") a small **software** has been developed by the author. Principal aim was to bring attention of students (future managers) on extended considerations of construction management especially at executing large-scaled and complex development works.

Due to standard technologies and to specialization of resources (subcontractors) also in Construction Industry some aspects of the problem of harmonyzing preferences of Clients and those of Contractors can be demonstrated by the challenge of Flow-Shop Problems. Expectation of completing products (buildings) as individual deliveries in the shortest times of execution (Clients' interest) is not necessarily coinciding the endeavour of contributing firms in completing series of products in the shortest overall execution time (Contractors' interest). Sometimes changing conventional/traditional order of operations in a technology may result in significant savings in time (that is in costs and in series of other efforts).

In a basic Flow-Shop Problem (mostly referred in manufacturing industry) *n* pieces of products are to be scheduled for production using uniform technology on *m* pieces of machines for to achieve the shortest overall execution (completion) time (C^{max}), or the minimum of sum of waiting times ($\Sigma\Sigma\delta_{ij}^{min}$) or some other target functions. The only differing feature in construction is that action of „machines" (subcontractors) can be overlapped in time at processing the same product (building).

Analogy between manufacturing industry and construction industry – from this later aspect – is trivial when considering numerous product-series to be manufactured and/or more buildings/structures to be built using the same „general" technology.

Main challenge is that for to solve most of the Flow-Shop Problems there exist no any close formula. For to find real optimal solution some type of enumerative algorithm is needed, which may/would take enormous run-time. Computations of those kinds are referred as „NP-hard" in technical literature, that is time (steps) needed for solution is „non-polinomial", it can not be defined as *n* times something, or n^2, n^3, etc., where *n* represents the number of products to be scheduled.

To demonstrate the difficulties with these calculations you can imagine a computer calculating one million master schedules – as permutations (sequence variants) of 20 projects – in a second. Well, to examine all possible permutations that computer should work for more than 77 thousand years ($20!=2.43\cdot10^{18}$; $2.43\cdot10^{12}$ sec > 77,000 years). So long a time for to solve a single problem usually we do not have. ...

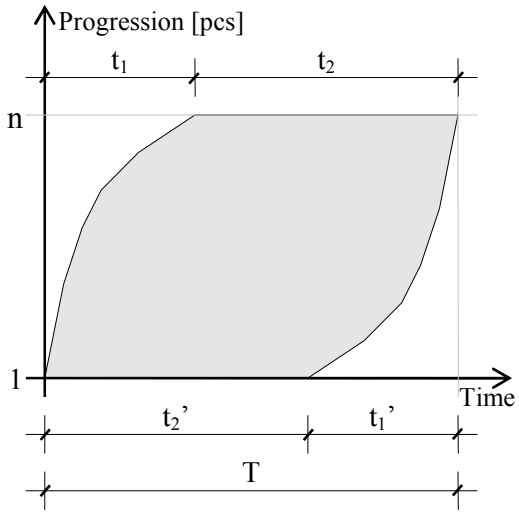
PROBLEMS TESTED

Modelling the problem

As indicated in the Preface linking up series of projects in an overall master schedule can be derived to the problem of matching pairs of succession vectors originated from most compressed „own schedules" of individual projects. Tracking necessary releases of own schedules during matching them in a master schedule estimates can be made on still available shortest overall execution time (and on minimum sum of necessary releases) avoiding the need of building up the sequences (permutations) totally for judgement. Thus huge sets of possible solutions can be eliminated from examinations.

Figure 3.

Constant (t_1) and variable (t_2) parts of overall time of completion (T) in a master schedule



$T = t_1 + t_2$

$T = t_1' + t_2'$

$t_1 = const$

$t_1' = const$

$T_{min} \Rightarrow t_{2\ min}$

$T_{min} \Rightarrow t_2'_{min}$

Before discussing data and algorithms some general recognitions (remarks) are worth to be explained highlighting common manners of variants of problems examined.

- Overall execution time of any master schedule can be divided in two segments:
  1., overall duration of processing on the first machine (t_1), that is sum of processing times on the first machine, which is constant value, not varying by the sequence of pieces;
  2., time span between finishing the first and finishing the last process on the last workpiece, that is actually effected by the sequence.

- Building up a master schedule from the very last or from the very first workpiece is a symmetrical problem, so the division of above can also be made as overall duration of processing on the last machine (t_1') and time span between starting the first and starting the last process on the first workpiece (t_2'). Due to didactical reasons we choosed the former division. (See: Figure 3.)

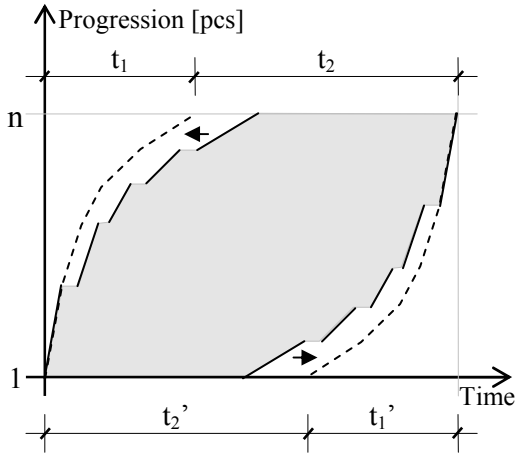
- In an optimal sequence (master schedule) the t_2 and t_2' values are at their minimum so these are the quantities the examinations should focuse on. Thus calculations are dealing with succession times, while processing times (durations) do have less or indirect importance.

- Expectation of non-overlapping can be handled as special case of overlapping as described in Preface, so overlapping is a more general expectation. (See: Figure 1.)

- Expectation of breakless performance of processes on the first and on the last machine has no significance from the point of view of the overall execution time, hence moving item lines (linear representations of processes) to the direction of start (on the first machine, in case of a broken performance) would have no effect on the overall execution time, while the performance can be made breakless. Similar can be said in case of broken performance on the last machine, moving item lines to the direction of the finish. (See: Figure 4.)

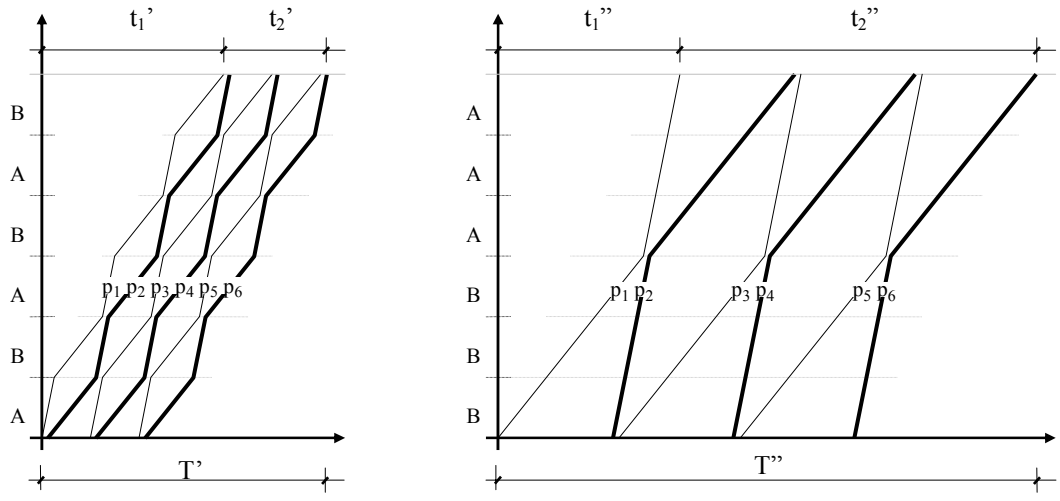
Figure 4.

Broken performance on the first and on the last machine can be made breakless without disturbing the overall execution time



Before paying more efforts to the problem we were interested in the range of effect of sequence on the overall execution time. For example, measuring theoretical schedules against each other when breakless performance of machines is expected we concluded that rate of overall execution time in an advantageous („best") schedule (T') and that in an unsuccessfully selected („worst") schedule (T") may even tend to zero(!) when increasing number of workpieces and that of machines. (See: Figure 5.)

Figure 5. Measuring effect of sequence on overall execution time with expectation of breakless performance of machines. (Sequencing *m* pieces of projects of type „A" and *m* pieces of projects of type „B" – all produced by *n* pairs of machines. On project typed „A" each second process has the duration of D_a, while others do have D_b. On project typed „B" the values of durations are the opposite. Values of critical approaches are CR. … Let D_a be significantly larger than D_b and CR! → $D_a >> D_b$,CR.)



$$\frac{T'}{T''} = \frac{t_1'+t_2'}{t_1''+t_2''} \approx \frac{m\cdot D_a + (n-1)\cdot D_a}{m\cdot D_a + m\cdot n\cdot D_a} = \frac{(m+n-1)\cdot D_a}{m\cdot(n+1)\cdot D_a} = \frac{m+n-1}{m\cdot(n+1)}$$

Based on overall durations of processes (on minimum timespan of all jobs assigned to each individual machine) and on most compressed own schedules of workpieces (projects) we also elaborated an algorithm for to estimate „theoretically" possible minimum (optimum) of target value (C^{max}). This later proved to be far more useful than expected at judging if a sequence (permutation) of larger number of workpieces (projects) produced or found is optimal or not.

The input

Data set
- number of workpieces (projects): 2 – 20 pieces (pcs)
- number of machines (processes): 2 – 20 pcs
- processing times (durations): 0 – 20 time unit (t.u.)
- critical approach (technological break): 0 – 10 t.u.
- number of random samples: 10^0 – 10^{10}

For to generate basic set of data random generator is built in. Durations and critical approaches can further be edited (modified) graphically, while edition of set of data (such as adding, deleting, copying projects and processes, and arranging data base) is provided via menu commands and hot-keys.

Target functions
- C^{min} („C^{max}" – minimum of overall execution time)
- D^{min} („$\Sigma\Sigma\delta_{ij}^{min}$" – minimum of sum of releases of own schedules)

Primary target function of the two above can be choosed optionally, while the other – as secondary target function – can be used for further filtering of optional solutions

Shop options
- overlapping (in time on a workpiece) allowed or not
- break of performance (of machines) allowed or not
- missing processes (on individual workpieces) allowed or not

Run options
- filter value (for enumeration) set manually
- filter value (for enumeration) adopted from random sampling
- run until first match (run until first sequence fitting the preset filter value is found)

Track options
- footprint of job (tracking graphically the run of enumerative algorithms along the combinatorical tree, and/or tracking pair-wise exchanges)
- distribution (tracking distribution of target values and/or progress of calculations)

Graphical options
- size and scale of graphical representation (cyclogram) of individual projects and consequently also that of the master schedule (for to provide best and overall view)
- indicating bottlenecks (succession times at their minimum), such as critical path at network techniques. (Here we name it as „knotwork" due to appearance of graphics resembling the hand-made artycraft decoration widely known as „macrame".)

The algorithms

Variants of enumeration
- total enumeration: testing all possible permutations
- partial enumeration: testing all permutations fitting preset filter value
- implicite enumeration: testing permutations fitting the actual filter value which is updated when better target value then actual filter is found

Variants of pair-wise exchanges
At pair-wise exchanges pairs of projects within the actual sequence (permutation) are tested for exchange systematically if the target value could be improved.
- exchanges generated by primary target function (if decreasing can be gained)
- exchanges generated by secondary target function (if primary target value is unchanged but decreasing can be gained in secondary target value)
- exchanges generated by secondary target function (if primary target value is unchanged but increasing can be gained in secondary target value)

Arranged Branch & Bound
Similar to implicite enumeration, but with no initial filter and with pre-testing and pre-arranging options/choices (branches of decision/combination tree) each time when proceeding downward along the decision tree (when adding newer and newer member to the sequence/permutation under construction).

Johnson's (1954) algorithm
Johnson's famous algorithm for to solve $F2//C^{max}$ problem (scheduling on two machines) published in 1954, adapted and tested for more than two machines

Random sampling
Numerous random-generated permutations tested and evaluated

Estimating optimum
Based on overall durations of processes (all jobs for each machine) and on most compact „own schedules" of projects (workpieces) „theoretical" minimum of target value (C^{max}) is estimated. Finding any permutation (either random-generated or gained via pair-wise exchanges) matching the estimates optimality of solution can be taken as granted for sure. Feature of estimating optimum is associated with random sampling.

EXPERIENCES

Enumerative algorithms (with apropriate filter values) are surely serving an or all the optimal solutions. First of main key-points is if there exists any optimal solution with target value fitting the filter, while choosing a filter value significantly worse than the real optimum may result in either exponentially increased run-time. Thus any failure in choosing/setting real and proper initial filter value has great significance. Second key-point is verification of optimum. It is an age-old experience of working with enumerative algorithms that solutions later proved to be optimal may appear quite fast during the calculations while to prove optimality of them (to be sure of not finding better ones) may take significantly longer run-time. It is the key-point where a proper estimate on the real optimum (as theoretically proved lower bound) can help much.

Nevertheless, run-time of enumerative algorithms are desparately determined by the data themselves under examinations. The higher the level on the combinatorical tree the algorithms (filters) can bound (stop) examination of improper branches the shorter the run-time is. This later proves to be in close relation with span, median and modus of distribution of target values of all the permutations. (How outstanding the optimum is, what proportion of all the permutations are optimal, etc.)

Pair-wise exchanges are fast and clear procedures and may result in a quite acceptable solution in an almost unmeasurably short run-time. Main disadvantage of them is that they may run to a local optimum. Thus initial sequence (permutation) the exchanges are started from has great significance. Combining these algorithms with random sampling for to set or choose initial sequence is a promising trial. Pair-wise exchanges raise a theoretical question too: It is admitted that any sequence can be transformed to any other one via pair-wise exchanges. The question is, if there exists any optimal sequence could be originated from a given non-optimal one via series of pair-wise exchanges while target values are monotonously improving. (If some secondary target function could help avoiding local optimum and could help finding global one.)

Arranged Branch & Bound suffers no risk of non-existing target value set as filter. It always starts from and results in existing solutions. Effort of arranging (pre-selecting) choices for building up sequences on each level downward the decision tree however consequences in a significantly increased run-time, while in finding the real optimum (either the first one) in a shorter run-time it hasn't proved. It may keep searching on branches (on subsets) later proved to be not optimal for long. (It was one of the most unexpected and unpleasant experiences of all the examinations. We assumed and expected more of that. Locally advantageous matching – pairing or building sub-sequences – of own schedules does not necessarily mean advantage in global.)

Johnson's (1954) algorithm for to solve two machine flow-shop ($F2//C^{max}$) problem surely results in optimal solution (for two machines), optimality of which can be theoretically proved. With some slight modification the algorithm can be adapted for overlapped situations too. The problem with the logic of the algorithm is that it proves to be false, even more counteracting, when applying for problems with more than two machines. Researches are going on to elaborate some rules and/or preferences to adapt it for more machines but one should pay not too much hope of success. (Contradiction of applied logic and of aimed target can also be proved theoretically.)

Random Sampling is the least developed – even more, it could be said, the most primitive – way of finding acceptable or hopefully optimal solutions, but it works suprisingly well. Main question is how representative the sample of trials can be considered. Increasing the number of pieces (projects) to be scheduled even seemingly huge numbers (either millions or billions) of samples tend to be less and less in proportion compared with the number of all possible sequences. But, does it really mean, as objective law, that sampling is less representative? To assist answering – or at least judging – this later question was the primary reason for features of Tracking Footprints and Distributions (together with routines of Estimating Optimum) had been built in the **software**.

Routines of Estimating the optimum has been developed originally for to cool over-heated expectations against time-savings via finding a proper sequence of pieces to be

scheduled. Later it proved to be a useful „benchmark" to judge optimality of any solution (sequence/permutation) found or produced. Main idea of estimate is that „t_2" segment of overall execution time of a master schedule on one hand can not be less than sum of succession times either at finish (Σf_i) or at start (Σs_i), respectively, on the most compressed project (t_2 value on the own schedule of which is the smallest). On the other hand, t_2 segment of total execution time (C^{max}) of a master schedule also can not be smaller than t_2 value of a fictive „overall" project (workpiece), durations of processes of which are calculated as overall time-span of processing on each machine, and considering the smallest approach times (CR) between succeeding processes.

This later has been improved to a test of selecting projects/workpieces systematically to the first and to the last position of the sequence using their actual critical approach values for calculating succession times, while durations were calculated from overall processing times for each machines – considering actual expectations of breakless performance. These estimates may include some errors (inaccuracy) anyway due to actual variancy of time values and of successions times. But during test-runs they proved to be correct so many a time that it drove our attention and interest to pay more efforst for searching ways of estimating optimum as correctly as possible. The bounty is alluring: establishing a „criterion of optimum". Though trials are promising in case of estimating C^{max} (t_2^{min}) values, similar „accuracy" still had not been achieved at estimating minimum of sum of necessary releases ($\Sigma\Sigma\delta^{min}$).

Window of displaying Footprints of calculations had been integrated into the software mainly for testing random generator routines (if the sample spans the whole range of possible sequences) and for to evaluate effectivity of enumerative routines (what percentage of possible sequences have been excluded from the examinations). The feature proved to be useful also for to cool expectations against enumerative approaches. Huge figures may be glamorous but misleading. Having the proper filter value not rarely we got the result of excluding more than 99.99999999 % of possible sequences. That is, the algorithm had to examine less than 0.00000001 % of all the permutations. But how long a time would it mean in case of sequencing let's say only 20 pieces (projects/workpieces)?

Another return of footprints were the ability of tracking the positions of individual pieces within the sequence while improving target value(s) via pair-wise exchanges. It was also good for destroying any preliminary assumptions and expectations or any theoretical rules we guessed.

Tracking Distributions of target values during run-time and after the calculations was also useful and awakening. On one hand the shape and extents of frequency-diagram furnished us some kinds of intimate information about behaviour and progress of calculations, on the other hand it warned us that limits of displaying result may obscure even huge numbers. Sometimes to an optimum value of a target function with a seemingly faint chance of finding (occurence) associated enormous number of alternative solutions. That is: a target value with relative frequency of seemingly zero may be achieved in huge number of cases (instances/permutations). It gives us some hope of random samples might prove to be representative in cases of larger number of projects (workpieces) too, and combined with other routines of improvement, they may lead us to find the real – or at least „acceptable sub-" – optimum.

SUMMARY

The problem of sequencing – and within that the so called Flow Shop Scheduling Problem – got into our view at the dawn of age of more and more spreading and available micro computers nowadays called PC-s. Historically the question we faced was if micro computers could be used for bringing still unavailable tool of sequencing closer to practice for to assist resolving some contradictions of interest of clients and that of contractors in a way of finding proper arrangements/sequences of contracts. (If each building could be delivered in an acceptably short period of disturbing the sites and performance of contractors contributing could also be managed on an „effective industrialized" way – in times of serious lack of building capacities and of mass and urgent need of (re)constructions.)

After modelling and testing potential effects of sequences on total execution time of a master schedule we tested five principal ways of building/finding optimal sequences:

1. For to gain certain optimum, and to check any other trials, enumerative algorithms had been developed, later improved and accelerated by some methods of filtering (Total-, Partial- and Implicite Enumeration);
2. Building sequence as a kind of string of optimal matches of individual schedules with the hope of deriving the problem back to a kind of Assignment Problem that can be solved either by Linear Programming („Arranged" Branch & Bound);
3. Finding partially optimal solutions for simpler cases and extending/combining them for more complex situations. That is, adapting some similar and theoretically provable algorithm to our purposes (Johnson's Algorithm);
4. Producing an initial sequence and improving it gradually via series of consecutive modifications (Pair-wise Exchanges).
5. Finally, for testing/measuring return of all our efforts against, a pure and primitive way of finding optimal sequence – „by chance" (Random Sampling).

Due to our principal aim of testing sequencing/scheduling as a tool for resolving some contradictions in Construction Industry and to get „provably sure" optimum, manyfold heuristics had been deliberately excluded from our investigations.

After long times of examinations, after numerous trials and hypotheses falling apart as leaves from the trees none of the principal ways above proved to be either the only or the best way of constructing/finding „the optimal" sequence. None of the „advanced" techniques and/or approaches proved to be either deliberately or more outstandingly better or effective for our purposes than the most primitive way of Random Trials. But the same time we found that elaborating a proper estimate on likely optimum is more promising a challenge. Having it, we can judge optimality of any sequence found or produced, and we can judge likely return of our efforts to find an even better solution if the one present did not seem to be optimal. ...

Construction Management, in general, avoids the „flow-shop problem" anyway. There are individual clients with diverse interest, there are contracts determining sequence of buildings and there are age-old traditional orders of processes within wide-spread and applied technologies. Sequences may be dictated by practical reasons of logistics and/or availability of resources of any kind. Risk of any changes in durations during

execution is also high, that may vanish all the results of our efforts spent for finding the optimal sequence. The question of sequencing emerges in case of – 'Thank God' – rare situations of need of mass reconstruction and/or in case of serious lack of capacities for delivering in a proper time. Main yield of dealing with it may appear rather in manufacturing construction materials and prefabricated elements, but most of all, in bringing attention of managers on the phenomenon that changing age-old traditions in technologies and/or selecting a proper sequence of jobs may result in unexpectedly considerable savings in time and/or in any other efforts. ...

REFERENCES

[1] John W. Brown, Donald R. Sherbert, *Methods of finite mathematics*, John Wiley & Sons, Inc. 1989

[2] Frank S. Budnick, Dennis McLeavy, Richard Mojena, *Principles of Operations Research for Management*, IRWIN Homewood, Illinois, 1988

[3] *Deterministic and stochastic scheduling, Proceedings of the NATO Advanced Study and Research Institute on Theoretical approaches to scheduling problems* held in Durham, England, 1981, ed. by M. A. H. Dempster, J. K. Lenstra, A. H. G. Rinnooy Kan

[4] *Symposium on the theory of scheduling and its application*, ed. by S. E. Elmaghraby, Berlin-Heidelberg-New York, Springer, 1973

[5] Wlodzimierz Szwarc, *Elimination methods in the m·n sequencing problem*, Naval Research Logistics Quarterly 18, PP. 295-305 (1971)

[6] Wlodzimierz Szwarc, *Optimal elimination methods in the m·n flow-shop scheduling problem*, Operations Research 21, PP. 1250-1259 (1973)

[7] Wlodzimierz Szwarc, *Dominance conditions for three-machine flow-shop problem*, Operations Research 26, PP. 203-206 (1978)

[8] T. Török - Z. Vattai, *Sequencing of Tasks with Identical Work Processes for Achieving The Shortest Possible Time of Realization*, PRIODICA POLYTECHNICA, ARCHITECTURE Vol 32. Nos. 3-4, PP. 195-207, Budapest, 1988

[9] Zoltán A. Vattai, *Applying Branch & Bound techniques for scheduling problems in Construction Industry*, dr. univ. theses (in Hungarian), 1993

[10] Zoltán A. Vattai, *A Descriptive Proof for Johnson's Algorithm for Solving Two-machine Flow-Shop Problem*, PERIODICA POLYTECHNICA SER. CIVIL ENG. VOL. 37, NO 3, PP. 249-260, Budapest, 1993

[11] Zoltán A. Vattai, *An algorithm for to solve F2/overlap/C^{max} problem*, Papers of Applied Mathematics vol 17, PP. 241-257, Budapest, 1993

[12] Zoltán A. Vattai, *Applying combinatorical models in construction management*, C.Sc. (Ph.D.) theses (in Hungarian), 1999

[13] Zoltán A. Vattai, *Scheduling Construction Of A Large-Scale Water-Proof Reinforced Concrete Foundation Slab*, International Conference On Technology And Management In Construction, Moscenicka Draga, 2003

ANNEXES

Figure 6. Found optimum and Distribution of target values at Scheduling 20 projects (Schedule displayed with option of Show Knotwork on)

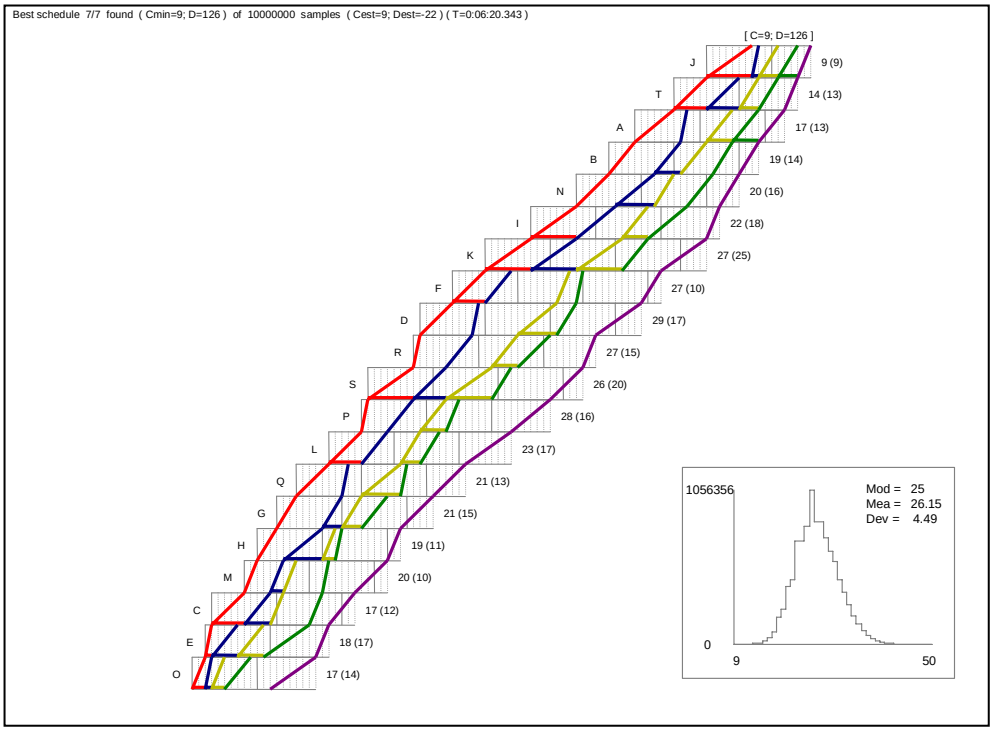


Figure 7. Footprint (on the decision/combinatorical tree) of an Implicite Enumeration of scheduling 9 projects/workpieces