dr inż. Marek Mika,
dr inż. Grzegorz Waligóra,
prof. zw. dr hab. inż. Jan Węglarz
Instytut Informatyki Politechniki Poznańskiej

METAHEURISTIC APPROACH TO THE MULTI-MODE RESOURCE-CONSTRAINED PROJECT SCHEDULING PROBLEM WITH DISCOUNTED CASH FLOWS AND PROGRESS PAYMENTS

Abstract. In this paper the multi-mode resource-constrained project scheduling problem with discounted cash flows is considered. A project is represented by an activity-on-node (AoN) network. A positive cash flow is associated with each activity. The progress payment model is considered and the objective is to maximize the net present value of all cash flows of the project. Local search metaheuristics: simulated annealing and tabu search are proposed to solve this strongly NP-hard problem. An extensive computational experiment is described, performed on a set of standard problems constructed by the ProGen project generator, where, additionally, the activities' cash flows are generated randomly with the uniform distribution. The metaheuristics are computationally compared, the results are analyzed and discussed and some conclusions are given.

1. INTRODUCTION

In the classical resource-constrained project scheduling problems (see e.g. [1] for a survey) the project duration (or makespan) is to be minimized. The multi-mode problem (MRCPSP) is a generalized version of the standard problem (RCPSP), where each activity can be executed in one of several modes representing a relation between resource requirements of the activity and its duration. The schedule has to be precedence- and resource-feasible and no activity may be interrupted. The resources can be renewable, non-renewable and/or doubly constrained, where renewable resources are limited period-by-period, non-renewable resources are limited for the entire project and doubly constrained ones are limited both for each period and for the entire project. The objective is to find an assignment of modes to activities as well as precedence- and resource-feasible starting times of all activities such that the makespan of the project is minimized. The problem is strongly NP-hard being a generalization of the RCPSP. The RCPSP is strongly NP-hard as a generalization of the well-known job shop problem.

Financial aspects of the problem appear when, generally, a series of cash flows (positive and/or negative) occur over the course of the project (see e.g. [4]). Positive cash flows (i.e. cash inflows) correspond to payments for the execution of activities. Negative cash flows (i.e. cash outflows) include expenditures for labour, equipment, materials etc. Time value of money is taken into account by discounting the cash flows. The most commonly considered financial objective is the maximization of the net present value (NPV) of all cash flows of the project. The resulting problem is then called the multi-mode resource-constrained project scheduling problem with discounted cash flows (MRCPSPDCF).

In real life situations there are at least two parties involved in the project: the client, who is the owner of the project, and the contractor, whose job is to execute the project. They have to agree the method of payment transferring from the client to the contractor for the execution of the project. The ideal situation for the client would be, of course, a single payment at the end of the project. The contractor, on the other hand, would like to receive the whole payment at the beginning of the project. It should be stressed that in this work we look for solutions maximizing the net present value from the contractor's point of view. There are several types of contracts and payment models considered in the literature. For a comprehensive survey see e.g. [2].

In this paper we deal with the MRCPSPDCF, where there are only positive cash flows and they are associated with activities, more precisely, a cash inflow (i.e. a payment going from the client to the contractor) is associated with the execution of each activity. This assumption corresponds to a common situation, where the contractor's expenditures do not depend on time but are fixed independently from the project's duration, or they are simply covered by the client. As a practical example, we can consider a large software project realized on the client's order (e.g. bank, insurance company etc.), implemented according to software engineering requirements. A software company (the contractor) has its own hardware (computers), software (operating systems, editors, compilers) and programmers, and bears only fixed expenses connected with its business activity. These expenses do not depend on the realized project. Another example could come from building engineering. A private investor (the client) places an order of house building in a building company (the contractor). The company has its own workers, specialized equipment, transport facilities etc. The expenses of all materials, energy, fuel and so on are covered by the client. Also in such a case, the contractor's expenditures do not depend on the project. Many similar examples from other areas can be given as well.

We consider a payment model which belongs to those of special interest in practice – progress payments (PP). In the ideal situation for the client, he makes a single payment to the contractor only at the end of the project. However, in general, this shifts the entire financial burden on the contractor, which may not be acceptable in some project environments. In consequence, the two parties often attempt to negotiate a method of payments over the duration of the project, e.g. according to the PP model. In this model payments are made at regular time intervals over the course of the project, where the client and the contractor agree about the length of this interval. A typical example is a contract, where at the end of each month the contractor receives his payment for the work accomplished during that month. It should be stressed that we assume the budged of the project to be predetermined, i.e. the contractor's total payment is known a priori and equal to the sum of all cash flows associated with the project's activities. As a result, the activities' cash flows do not depend on their execution modes, since the client pays the same regardless of how the contractor realizes the project.

Since, as we have mentioned, the problem is strongly NP-hard, we propose local search metaheuristics: simulated annealing (SA) and tabu search (TS) to approach it. The paper is organized as follows. In Section 2 we define the considered problem mathematically. In Section 3 the applications of the metaheuristics: SA and TS are described. Section 4 is devoted to the computational experiment and the analysis of the results. Finally, some conclusions and directions for further research are given in Section 5.

2. PROBLEM FORMULATION

We consider the multi-mode resource-constrained project scheduling problem with positive discounted cash flows (MRCPSPDCF) which can be formulated as follows. A project consisting of n activities is represented by an activity-on-node (AoN) network $G = (J, E)$, $|J| = n$, where nodes and arcs correspond to activities and precedence constraints between activities, respectively. No activity may be started before all its predecessors are finished. Nodes in graph G are numerically numbered, i.e. an activity has always a higher number than all its predecessors. Each activity j, $j = 1,...,n$, has to be executed in one of M_j modes. The activities are non-preemtable and a mode chosen for an activity may not be changed (i.e. an activity j, $j = 1,...,n$, started in mode m, $m \in \{1,...,M_j\}$, must be completed in mode m without preemption). The duration of activity j executed in mode m is d_{jm}. A positive cash flow CF_j is associated with the execution of each activity j. We assume that there are R renewable and N non-renewable resources. The number of available units of renewable resource k, $k = 1,...,R$, is R_k^ρ and the number of available units of non-renewable resource l, $l = 1,...,N$, is R_l^ν. Each activity j executed in mode m requires for its processing r_{jkm}^ρ units of renewable resource k, $k = 1,...,R$, and consumes r_{jlm}^ν units of non-renewable resource l, $l = 1,...,N$. We assume that all activities and resources are available at the start of the project.

When dealing with the NPV criterion time value of money is taken into account by discounting the cash flows. In order to calculate the value of NPV, a *discount rate* α has to be chosen, which represent the return following from investing in the project rather than e.g. in securities. Then the *discount factor* $(1+\alpha)^{-T}$ denotes the present value of a dollar to be received at the end of period T using a discount rate α.

The objective of the MRCPSPDCF is to find an assignment of modes to activities as well as precedence- and resource-feasible starting times for all activities such that the net present value of the project is maximized.

In the progress payment (PP) model the client makes the payments to the contractor at regular time intervals until the project is completed (e.g. at the end of each month the contractor may receive a payment for the work accomplished during that month). The last payment is made at the completion of the project. Since the length of the time interval between payments is agreed and does not depend on the project's duration, the number of payments is not known a priori. The value of NPV is given by the following formula:

$$NPV = \sum_{p=1}^{H-1} P_p (1+\alpha)^{-p \cdot T} + P_H (1+\alpha)^{-C_{max}} \qquad (1)$$

where P_p is the payment at payment point p, $p = 1,2,...,H$, T is the assumed interval length, C_{max} is the completion time of the project (i.e. the project's makespan) and H (the number of payments) is the smallest integer greater or equal to C_{max}/T.

In order to compute the payments P_p at successive payment points, partial cash flows (linearly proportional to activities' durations) of all the activities executed in the considered interval (i.e. since the last payment point) are summed up.

3. METAHEURISTICS

3.1. Common features

3.1.1. Solution representation

A feasible solution is represented by two n-element lists. The first one is a precedence-feasible permutation of activities, in which each activity j, $j = 1,...,n$, has to occur after all its predecessors and before all its successors. This structure is called the *activity list*. The second one is a list of execution modes for all activities and is called the *mode assignment*. The j-th element of this list defines the execution mode of activity j.

The starting times of all activities are defined by using the serial SGS (Schedule Generation Scheme) decoding rule. Since under positive cash flows the NPV criterion is a regular performance measure [4], i.e. is non-decreasing in activity completion times, we may use the serial SGS rule to construct the schedule.

3.1.2. Preprocessing

In order to reduce the search space and adapt the project data to the implementations of the metaheuristics, the preprocessing procedure introduced by Sprecher et al. in [7] is used. After the execution of this iterative procedure, all non-executable modes, inefficient modes as well as redundant non-renewable resources are deleted.

3.1.3. Objective function

When calculating the objective function, we use a penalty function whose value depends on how much a solution exceeds the non-renewable resource constraints. To this end, for each solution s we calculate a *Solution Feasibility Test* function $SFT(s)$ according to the following formula:

$$SFT(s) = \sum_{l=1}^{N} \max\left\{0, \sum_{j=1}^{n} r_{jml}^{\nu} - R_l^{\nu}\right\} \tag{2}$$

Solution s is feasible with respect to non-renewable resources if and only if $SFT(s) = 0$.

Consequently, we define the penalty function for each solution s as:

$$NPV(s) = \begin{cases} \text{formula (1)} & \text{if } SFT(s) = 0; \\ \sum_{j=1}^{n} CF_j (1+\alpha)^{-[TH+SFT(s)]} & \text{otherwise} \end{cases} \tag{3}$$

where TH is the time horizon of the project (the upper bound of the project's makespan given by the sum of maximal durations of activities).

3.1.4. Starting solution

A starting solution for each instance is generated by setting all activities on the activity list in an ascending order that follows from the ordering of nodes in the precedence relation graph, and executing all jobs in their first modes.

3.1.5. Stop criterion

The stop criterion has been defined as an assumed number of visited solutions.

3.2. Simulated Annealing

3.2.1. Neighbourhood

Neighbour solutions are generated from the current one using one of the following three operators:

- *activity shift* – operates on the activity list (*AL*) in the following way:
  - one activity j is randomly chosen on AL;
  - the nearest predecessor p and the nearest successor s of activity j are found on AL;
  - a position x between activities p and s is randomly chosen;
  - activity j is moved to position x.
- *mode change* – operates on the mode assignment (*MA*) in the following way:
  - a position y on MA is randomly chosen;
  - the mode in position y is changed to another one randomly chosen, if possible.
- *combined move* – operates simultaneously on both AL and MA and is a composition of the activity shift and the mode change.

For each transition the above-mentioned operators are chosen randomly with a certain probability which depends on the instance of the considered problem. The probability of a combined move is set at a fixed value of 0.1. The probability of an activity shift depends on the density of graph G and is given by the following formula:

$$P_{AS} = \left(\frac{1}{2} - \frac{|E| + |A|}{n \cdot (n-1)} \right) \tag{4}$$

where $|E|$ is the cardinality of the set of precedence constraints (i.e. the set of arcs of graph G) and $|A|$ is the cardinality of set A of arcs of the transitive closure of graph G. The probability of the mode change $P_{MCh} = 0.9 - P_{AS}$.

3.2.2. Cooling scheme

The adaptive cooling scheme known as "polynomial-time" is used to control the cooling process of the SA algorithm. The only exception introduced into this implementation of SA is the stop criterion, which is set at a fixed number of visited solutions. For more details concerning our implementation of SA for the MRCPSPDCF see [3].

3.3. Tabu Search

3.3.1. Neighbourhood

Similarly as it was for simulated annealing, neighbours of the current solution are generated by swapping activities on the activity list and changing modes on the mode assignment. More precisely, in the case of TS there are two operators used:

- *activity swap* – this operator swaps each two activities on the activity list that may be swapped without violating the precedence constraints; as a result, a set of neighbouring solutions is created each of them differing from the current solution in two positions of the activity list;
- *mode change* – this operator changes the execution mode of each activity to every other executable mode for this activity; as a result a set of neighbouring solutions is created each of them differing from the current solution in one position of the mode assignment.

3.3.2. Tabu list

The tabu list is managed according to the *Tabu Navigation Method*. There is only one list, so-called *Tabu List* (*TL*), which is a queue of a given length. Whenever a move is performed, its reverse is added to *TL* in order to avoid going back to a solution already

visited. The oldest existing move is removed from the front of the list (according to the FIFO policy). All moves existing on *TL* are tabu. However, the tabu status of a move may be cancelled according to so-called *aspiration criterion*. It allows the algorithm to move to a tabu solution (perform a tabu move) if this solution is better than the best found so far. The important parameter in the TNM is the length of *TL*. In our implementation it is set at 7.

3.3.3. Restarting

In case there are no admissible solutions in a neighbourhood, the algorithm has to restart the search process in order to visit an assumed number of solutions. Additionally, the restarting mechanism is also used when an assumed number of iterations have been performed without improving the objective function. This number has been set at 20. If 20 iterations with no improvement are performed the algorithm restarts since further search in that solution region is considered hopeless.

In order to restart the search process from a different solution, it is generated randomly, i.e. the activity list is generated in the same way as described in point 3.1.4. but the mode assignment is generated randomly. For more details concerning our implementation of TS for the MRCPSPDCF see [3].

4. COMPUTATIONAL EXPERIMENT

In this section we present the results of a computational experiment concerning the implementations of the two metaheuristics for the considered MRCPSPDCF. The implementations were coded and compiled in C++ and ran on an SGI ORIGIN 3800 machine with 64 RISC R12000 processors and 57.6Gflops overall computing power, installed in the Poznań Supercomputing and Networking Center.

We used a set of benchmark problem instances from the PSPLIB library, generated using the project generator ProGen. We used benchmark sets for problems with 10, 12, 14, 16, 18, 20 and 30 non-dummy activities. All non-dummy activities may be executed in one of three modes. There are two renewable and two non-renewable resources. The duration of job j in mode m varies from 1 to 10. For each problem size a set of 640 instances was generated, but for some instances no feasible solution exists and therefore these instances were excluded from further consideration. For more details concerning ProGen and the PSPLIB library see [6].

For each instance from the PSPLIB library we have generated cash flows of all activities from the interval (0; 1000] with the uniform distribution.

In order to ensure a comparable computational effort devoted to both the metaheuristics, the stop criterion has been defined as a number of solutions visited. This number has been set at $12000 \cdot n$, where n is the number of activities.

In order to complete the specification of the instances, we need to define α – the discount rate per period. In our experiments we assume that one period corresponds to one month. The experiment was performed for five different values of the discount rate. We assumed that $\alpha \in \{0.01, 0.02, 0.03, 0.05, 0.1\}$. We also assumed different values of the interval length T. The selection of these values was based on the optimal values of the makespan for the instances used. For each number of activities, the makespan optimal solutions for the PSPLIB instances vary from 8 to 62. We assumed the payment

interval length $T \in \{3, 4, 6, 12\}$ in our experiment, which corresponds to payments made every 3, 4, 6 and 12 months. Of course, the cases where this length equals or exceeds the project's duration ($T \geq C_{max}$), lead directly the lump-sum payment (LSP) model, where a single payment is made at the completion of the project.

Below we present the results of the experiment. For each problem parameters the following numbers are shown:

- # – the number of instances for which the algorithm found a solution equal to the best solution known (i.e. best solution found by either of the algorithms);
- AAD – the average absolute deviation from the best solution known;
- MAD – the maximal absolute deviation from the best solution known.
- ARD – the average relative deviation from the best solution known;
- MRD – the maximal relative deviation from the best solution known;

The results of the experiment are presented in Table 1. Although we present the results for 10, 20 and 30 activities only, the analysis of the results has been performed for all considered values of n and the conclusions given below are general.

It can be noticed that TS performs better for smaller number of activities, whereas the efficiency of SA grows with the value of n, both in terms of the number of best solutions found and the average and maximal relative deviations. Nevertheless, the maximal relative deviations for SA can be quite large in some cases, which suggests that if the algorithm fails to find a good solution, its result may be quite distant from the one produced by TS. It should be stressed that the maximal deviations for TS are, in general, much smaller than for SA, which suggests that TS finds poor solutions very rarely. The growth of the discount rate α improves the results produced by TS in terms of the number of best solutions found. The relative and maximal deviations from the best known solution increase with the value of α for TS as well as for SA, but for SA they grow more rapidly. This, however, can be explain by the fact that although the value of NPV decreases with the growth of α, the differences in the value of NPV between two different solutions increase with the value of the discount rate.

As far as computational times are concerned, they vary from around 1.5 sec. for 10 activities up to 12-16 sec. (depending on the interval length T) for 30 activities, and are very similar for both the metaheuristics.

We can also see that shortening the interval length T improves the performance of TS, whereas SA is better when the payments are made less often. For larger numbers of activities, although SA outperforms TS, as it has been mentioned before, the growth of the number of payments leads to an evident improvement of the TS results, whereas there is a deterioration of the results obtained by SA. This fact confirms the observation that the more the problem approaches its makespan minimization version (by decreasing the number of payments), the better results are produced by SA.

5. CONCLUSIONS

In this paper the multi-mode resource-constrained project scheduling problem with positive discounted cash flows and the maximization of the net present value of all cash flows has been considered. The progress payment model has been examined, where payments are made at regular time intervals. Applications of two local search metaheuristics: simulated annealing and tabu search for the considered problem have been presented. These metaheuristics have been compared on the basis of a

computational experiment performed on a set of instances obtained from standard test problems constructed by the ProGen project generator, where, additionally, cash flows were generated randomly with the uniform distribution.

The results of the computational experiment show that, generally, the tabu search algorithm performs better for a smaller number of activities, whereas the growth of the number of activities improves the efficiency of the simulated annealing algorithm. On the other hand, under a fixed number of activities, TS gets the advantage over SA with the growth of the discount rate. TS also behaves better when payments are made more often, otherwise SA seems to be more effective. The more the problem approaches its makespan minimization version (by decreasing the number of payments), the better results are produced by SA. The above features of the proposed metaheuristics can be taken into account when dealing with a particular class of practical MRCPSPDCF problems.

In the future we plan to continue the research on the application of the metaheuristics to different MRCPSPDCF problems. In particular, it will be interesting to add negative cash flows (i.e. cash outflows) to the considered problem as well as to examine other payment models. Although we focus on the NPV criterion, we intend to analyze also other financial criteria. In the further research it is planned to construct a branch-and-bound procedure for the considered problem in order to find optimal solutions. Comparing the results of the metaheuristics with optimal solutions will allow to fully evaluate their efficiency.

ACKNOWLEDGEMENT

This research has been supported by the Polish State Committee for Scientific Research (KBN) grant no. KBN 4T11F 001 22.

REFERENCES

[1] P. Brucker, A. Drexl, R. Möhring, K. Neumann, E. Pesch, Resource-constrained project scheduling: notation, classification, models and methods, European Journal of Operational Research, 112, 1999, 3-41.

[2] W.S. Herroelen, P. Van Dommelen, E.L. Demeulemeester, Project network models with discounted cash flows: a guided tour through recent developments, European Journal of Operational Research, 100, 1997, 97-121.

[3] J. Józefowska, M. Mika, G. Waligóra, J. Węglarz, Mutli-mode resource-constrained project scheduling problem with discounted positive cash flows - simulated annealing vs. tabu search, Foundations of Computing and Decision Sciences, 27 (2), 2002, 97-114.

[4] A. Kimms, Mathematical Programming and Financial Objectives for Scheduling Projects, Kluwer Academic Publishers, Dordrecht, 2001.

[5] R. Kolisch, Project Scheduling under Resource Constraints – Efficient Heuristics for Several Problem Classes, Physica, Heidelberg, 1995.

[6] R. Kolisch, A. Sprecher, PSPLIB – a project scheduling problem library, European Journal of Operational Research, 96, 1997, 205-216.

[7] A. Sprecher, S. Hartmann, A. Drexl, An exact algorithm for project scheduling with multiple modes, OR Spektrum, 19, 1997, 195-203.

Table 1. Results of the experiment

			TS					SA				
n	α	*T*	#	AAD	MAD	ARD [%]	MRD [%]	#	AAD	MAD	ARD [%]	MRD [%]
10	0.01	3	521	0,20	19,66	0,00	0,47	422	10,17	329,42	0,21	5,03
		4	513	0,21	15,14	0,00	0,32	418	11,85	335,74	0,25	5,30
		6	524	0,11	12,66	0,00	0,31	426	8,39	214,61	0,18	4,11
		12	519	0,34	33,03	0,01	0,89	442	6,65	155,29	0,15	3,51
	0.02	3	524	0,37	55,08	0,01	1,81	397	22,90	584,33	0,53	10,05
		4	516	0,33	28,72	0,01	0,64	398	23,79	592,03	0,57	9,76
		6	525	0,19	26,81	0,01	0,84	413	18,12	523,08	0,44	10,12
		12	519	0,58	56,46	0,02	1,76	428	12,50	275,66	0,34	6,77
	0.03	3	524	0,48	69,85	0,01	2,59	392	33,22	715,73	0,86	14,26
		4	518	0,50	80,38	0,02	3,02	384	33,69	784,72	0,88	13,89
		6	527	0,17	33,86	0,01	1,15	397	25,41	679,13	0,70	14,34
		12	518	0,83	84,43	0,03	2,60	419	16,50	343,62	0,52	9,67
	0.05	3	524	0,48	83,63	0,02	3,86	359	52,96	941,05	1,62	21,30
		4	525	0,38	83,25	0,01	3,95	355	53,34	1027,87	1,68	20,89
		6	527	0,28	68,54	0,01	3,37	363	39,72	631,51	1,36	21,01
		12	521	0,60	84,15	0,03	3,98	394	22,10	485,09	0,91	19,94
	0.1	3	528	0,34	109,54	0,01	3,90	307	75,53	926,34	3,29	29,35
		4	529	0,34	76,56	0,01	2,94	307	71,46	1024,48	3,31	32,92
		6	527	0,28	84,67	0,02	7,06	318	58,81	903,15	3,08	33,18
		12	525	0,27	68,28	0,02	6,19	368	24,52	405,54	1,94	30,87
20	0.01	3	328	10,33	261,07	0,12	3,48	464	19,41	574,29	0,22	5,46
		4	322	10,45	307,74	0,12	3,62	466	18,79	523,44	0,21	5,17
		6	330	10,61	243,89	0,13	2,90	463	16,45	576,72	0,19	6,25
		12	312	13,24	224,80	0,16	2,99	478	11,27	494,22	0,13	5,49
	0.02	3	360	9,91	187,24	0,13	2,19	421	54,17	968,98	0,71	11,46
		4	372	9,02	200,81	0,12	2,09	423	53,80	971,60	0,72	11,23
		6	360	10,03	145,16	0,14	2,33	430	44,65	960,46	0,61	13,81
		12	355	13,64	300,90	0,20	5,55	443	33,13	850,09	0,48	11,66
	0.03	3	395	8,24	221,39	0,13	3,03	372	91,28	1364,12	1,36	20,05
		4	409	7,72	224,38	0,12	2,69	375	89,01	1202,55	1,35	17,71
		6	396	8,27	132,36	0,13	3,22	392	74,68	1243,98	1,17	19,22

		12	378	12,77	253,97	0,22	4,53	410	52,92	1067,08	0,90	16,45
	0.05	3	428	6,84	243,18	0,13	4,23	319	136,58	1775,48	2,47	25,99
		4	447	5,73	197,10	0,11	4,22	328	126,53	1364,34	2,40	25,23
		6	430	6,43	133,00	0,13	2,98	335	109,38	1253,93	2,19	25,51
		12	406	9,87	186,20	0,24	3,53	382	59,97	1114,10	1,44	21,57
	0.1	3	449	5,21	195,16	0,16	6,35	290	179,05	1779,46	5,11	42,47
		4	448	4,58	189,93	0,15	6,55	310	155,10	1474,64	4,72	41,59
		6	431	6,28	258,55	0,22	9,82	308	125,86	1311,05	4,38	41,07
		12	413	6,73	183,34	0,33	9,66	363	59,11	839,96	2,99	44,70
30	0.01	3	243	40,48	482,16	0,33	3,76	454	15,16	585,57	0,12	4,56
	"	4	244	38,72	448,55	0,32	3,52	453	16,52	628,67	0,13	4,90
		6	227	41,31	363,88	0,34	3,20	468	14,78	602,44	0,12	5,26
		12	228	49,62	429,07	0,42	4,23	495	9,00	466,26	0,08	4,05
	0.02	3	309	31,74	879,03	0,31	8,77	392	75,68	1555,85	0,72	13,67
		4	307	29,98	493,80	0,29	5,43	386	71,65	1501,40	0,68	14,00
		6	291	31,46	445,43	0,31	5,18	409	55,94	1266,09	0,55	12,29
		12	260	48,05	405,22	0,51	5,49	466	37,50	1127,98	0,40	12,27
	0.03	3	350	26,41	382,79	0,29	4,53	352	129,41	2124,02	1,41	21,43
	"	4	330	29,56	432,19	0,33	5,18	373	111,37	1525,45	1,23	16,08
		6	305	35,09	533,19	0,41	8,20	388	99,13	1591,42	1,14	19,94
		12	280	44,43	448,16	0,56	6,32	438	47,26	1073,20	0,60	12,71
	0.05	3	405	19,79	378,14	0,29	6,06	298	221,51	2036,94	3,10	29,18
		4	389	21,02	356,74	0,31	5,97	303	201,50	2049,55	2,91	26,91
		6	376	22,80	367,70	0,35	5,87	316	178,92	1903,28	2,74	27,37
		12	327	32,64	399,65	0,58	7,28	385	93,06	1515,43	1,71	25,99
	0.1	3	364	20,91	305,06	0,50	7,53	337	180,22	1912,77	4,23	41,81
		4	366	18,86	301,69	0,48	6,47	329	155,91	1958,09	3,98	46,28
		6	337	19,67	280,70	0,55	9,73	365	127,47	1601,52	3,56	40,74
		12	312	24,92	394,09	0,93	14,72	407	56,75	1092,08	2,31	44,07