Planning & Scheduling Benchmarks
Resource Constrained Project Scheduling

Introduction

This data set, and its associated series of problems, is based upon large scale assembly but is applicable to a wide variety of problems in engineering, construction, and manufacturing that can generally be described as resource constrained project scheduling. This same data set and series of problems could have originated from a commercial construction project, a petro-chemical plant start-up, an automotive assembly line set-up, Space Shuttle reprocessing, or the assembly of heavy machine tools. I have prepared a first series of problems based upon this data set. Time permitting, I will prepare additional problems that are based upon this same data set or prepare additional data sets that can be the subject of this same series of problems.

In brief summary, this series of problems is based upon the most common questions raised by production management. The capabilities demonstrated in the solution of these problems can be used to answer the following:

If there were no resource constraints, how long would this work take? (Problem 1)

If we had unlimited labor resources, how long would this work take? (Problem 2)

How much labor should we allocate in order to meet our delivery schedule? (Problem 12)

Given a fixed labor budget and a specific policy on how the work is to be managed, how long will this work take? (Problem 3 and 4)

How can we accommodate new work without perturbing existing plans and schedules? What is the earliest that I can complete the new work? (Problem 5) How long can I delay the new work? (Problem 6)

Things haven't gone exactly as planned. Is there any impact on our promised delivery? (Problem 7) Or do we need to call in some overtime (Problem 8)?

It looks like we need to support another project. Can we divert some labor without impact on our promised delivery (Problem 9) Or can we adjust our schedule and compensate with some overtime (Problem 10)?

Can we use some of our excess labor to start the production of another assembly without impact on our promised delivery of the current assembly? (Problem 11)?

What delivery schedule can we promise our customers when we go into regular production (Problem 12)?

You will notice that every problem in the first series requires the construction of a schedule, and that the primary focus seems to be on producing "optimal" schedules, but taken as a whole the first series of problems represents an entirely different sort of challenge. My experience is that less than 1% of the code in a working scheduling application is devoted to schedule optimization. The remainder of the code is devoted to a diverse collection of functions required to make the application useful and user-friendly (graphical user interface, postscript reports, etc.) By similar measure, less than 5% of the time spent scheduling is making new schedules while 95% of the time is spent revising and maintaining schedules in the context of daily progress and changing assumptions. Hence, the real emphasis in the first series of problems is upon having a complete collection of schedule construction and maintenance capabilities that can be used to directly support production management. These problems exercise a scheduling engine with many of the very same challenges that it would encounter if deployed in a real production environment.

Even though the emphasis is not upon optimization, it is still necessary to produce good solutions to these problems. When a customer has a choice of systems that provide equivalent functionality, they will always choose the one that produces better schedules. However each industry and application has a different measure of quality. In the following problems, the focus is on minimal cycle time but in the actual factory setting there are a large number of other, sometimes subjective, measures of quality that must be respected. These other factors are not part of this first series of problems only to keep its scope limited and focused.

When measuring the quality of a schedule, it is important to agree not only upon the metric for comparison but also upon the standard for comparing two metric results. Traditional statistical measures can be used effectively to determine when two scheduling methods consistently produce different results but this should not be confused with what end users perceive to be a significant difference. In this, and other similar applications, a day is the smallest significant time quantum. This is because so many other aspects of production, like the start of a new assembly or the delivery of parts, are aligned on day boundaries. Two schedules that differ by a few minutes or a few hours will not be significantly different. In such settings, a schedule must be at least 1 day better in order to be significantly better, but a 1 day difference can be worth $100,000 or even $1,000,000 depending upon the industry involved.

It will be noticeable that this first series of problems is devoid of any "interesting" constraints or situations. In this data set there are only two kinds of constraints, resource and precedence. And we explore only one kind of resource usage. At a later time I hope to publish additional data sets that explore other technical issues including interruption and preemption, placement preferences, resource modeling, and state constraints.

To some extent, the twelve problems given below are arranged according to progressive difficulty. Problem 1 is concerned only with precedence while Problem 2 is concerned with both precedence and resource constraints. But Problem 2 and Problem

3 are qualitatively quite different. Problem 2 can be solved with a scheduling engine that possesses no lookahead but Problem 3 requires sufficient lookahead to accurately schedule within shift boundaries. Problem 4 requires the ability to look past the shift boundaries so long as sufficient resources are available. Problem 5 requires the ability to look beyond shift boundaries and to perform scheduling within the context of some existing, but irregular pattern of resource availability. Problem 6 requires essentially the same capability but may prove to be prohibitively difficult for scheduling engines that perform purely chronological schedule construction. Problem 7 exercises a scheduling engine's ability to perform incremental schedule modifications, while Problem 8 exercises a system's ability to constrain its operations to specific windows of time. Problems 9 and 10 are similar but might prove difficult for systems that rely upon hard coded models of resource availability rather than explicit data representations. Problem 11 again looks much like Problem 4 but tests whether the scheduling engine can be constrained in its effects while still pursuing an optimization goal. And finally Problem 12 exercises the scalability of a scheduling engine and at the same time tests whether a scheduling engine can simultaneously optimize multiple objectives.

You will find that the problems in the first series require quite a variety of capabilities. It is not necessary that 100% of the work required to solve a problem be automated, but participants are expected to report on what parts of the solution process were accomplished through data manipulation and commands issued to their scheduling system. To be honest, the only thing that counts is the ability to produce good solutions to these problems in a cost-effective and timely fashion. Combinations of manual and automated methods will always be necessary to adapt to the continuously changing manufacturing environment.

Some participants will find the complete series of problems to be prohibitively difficult. Others will find them to be within their capability. In the end, the number and kind of solutions submitted will substantially influence perceptions of what constitutes the state-of-the-practice and the state-of-the-art. At the same time, the number and kind of solutions submitted will influence the level of difficulty in subsequent benchmark problems

I recognize that most participants will be reluctant to publish their results without some knowledge of what others have accomplished. To reduce such anxiety, my partner in this endeavour, Mark Ringer, has agreed to work the first 4 problems with a simple first fit, interval based scheduling engine, with no optimization. His results are summarized in the section on results submitted by participants. I hope that all participants will report as many results as possible so that we can begin to determine what is a reasonable level of capability in addition to what is an achievable level of capability.

The physical problem, which this abstract data set represents, is the process of creating a large assembly consisting of a large number of discrete steps. Each step entails the performance of specific work documented in formal process plans. We can assume that the activity name, consisting of an assembly number followed by a step number, uniquely identifies each step and provides an unambiguous link to the formal process plans. This data set contains only the information required to schedule the work and all other information necessary for the performance of the work is found elsewhere. The duration of each step is annotated in hours and minutes and the work has been divided so that no single step requires more than 1 shift (7.5 hours) for its performance. Each step requires zero or more individuals drawn from 4 labor pools, Px, Py, Pz, and Pw, to be present for the full duration. While performing a step these individuals and their supporting equipment may occupy or exclusively reserve space within designated work-zones (Za, Zb, etc.) around the assembly.

This data set was synthesized from actual assembly problems and, based upon my experience, is perfectly realistic. However the specific names, durations, labor, zone, and precedence constraints have all been created to make this problem entirely unique, and dissimilar from any real assembly. Hence, labor is named by anonymous pools Px, Py, Pz, and Pw, work zones around the assembly are named Za, Zb, etc., and the steps are designated by a two part name consisting of an anonymous assembly number and an anonymous step number. In addition, the step numbers have been randomized so that there is no correlation between step number and precedence.

The data set, consisting of four parts, contains all of the data required in the graduated series of problems described below. The format for the data was chosen not for its generality or expressiveness, but rather for its simplicity so that minimal effort would be required in the creation of readers and writers. And in order to further minimize the effort required to get started, we have provided C++ source code for routines that will read and write this data. In brief summary, section 1 of the data set contains the activity specifications consisting of name, duration, and resource requirements; section 2 of the data set contains the precedence relations between the activities specified in section 1; section 3 of the data set contains a feasible schedule for the performance of the activities specified in sections 1 and 2; and finally section 4 contains miscellaneous data found in the following problem descriptions. C++ source code that implements basic reading and writing routines for the data found in sections 1, 2, and 3 is provided in a separate file, courtesy of Mark Ringer.

The full data set for Resource Constrained Project Scheduling can be retrieved by accessing the Planning & Scheduling Benchmarks WWW home page with your favorite web browser (mosaic, netscape, etc.) at the following address

"http://www.neosoft.com/~benchmarks"

Alternatively, you can access the data files and problem descriptions via anonymous FTP at address:

"ftp.neosoft.com" in directory "/pub/users/b/benchmrx"

Or you can retrieve the data files, problems descriptions, and results submitted by participants by sending an e-mail message to:

"benchmrx@neosoft.com" containing either of the messages
"get rcps problems" or "get rcps submissions"

NOTE:

Any particpants that received prior beta copes of this data should destroy those old data sets and use only the data acquired from this source. There have been some significant changes since it was first created.

Data Format

Section 1 of this data set is a table with a regular row/column structure which contains the names, durations, and resource requirements for 575 activities to be scheduled Each row defines a schedulable activity. Each column defines an attribute of the schedulable activities as summarized below:

Column	Attribute	Type
1	Name	Dotted alphanumeric: assembly_number.step_nnumber
2	Duration	Time: hours:minutes
3	Labor.Px	Integer: number required during activity execution
4	Labor.Py	Integer: number required during activity execution
5	Labor.Pz	Integer: number required during activity execution
6	Labor.Pw	Integer: number required during activity execution
7	Zone.Za	Integer: number required during activity execution
8	Zone.Zb	Integer: number required during activity execution
9	Zone.Zc	Integer: number required during activity execution
10	Zone.Zd	Integer: number required during activity execution
11	Zone.Ze	Integer: number required during activity execution
12	Zone.Zf	Integer: number required during activity execution
13	Zone.Zg	Integer: number required during activity execution
14	Zone.Zh	Integer: number required during activity execution
15	Zone.Zi	Integer: number required during activity execution
16	Zone.Zj	Integer: number required during activity execution
17	Zone.Zk	Integer: number required during activity execution
18	Zone.Zl	Integer: number required during activity execution
19	Zone.Zm	Integer: number required during activity execution

Section 2 of this data set is a table with a regular row/column structure which contains the precedence relations between the activities to be scheduled. Each row is a pair that defines a predecessor-successor relationship between two of the schedulable activities. The first column contains predecessors and the second column contains successors. No attempt was made to include all of the implied relationships nor to exclude any of the implied relationships. All of the necessary relationships are present and some implied relationships may be present.

Column	Attribute	Type
1	Predecessor	Dotted alphanumeric: assembly_number.step_number
2	Successor	Dotted alphanumeric: assembly_number.step_number

Section 3 of this data set is a schedule for the performance of this work that satisfies all of the constraints prescribed in Problem Number 4 (described below). It is provided as a table with a regular row/column structure which contains activity names and start times. Each row names an activity in column one and specifies its start time in column two.

Column	Attribute	Type
1	Name	Dotted alphanumeric: assembly_number.step_number
2	Start Time	Day/Shift+Hours:Minutes

In all cases, activity durations are expressed in hours and minutes in the format HH:MM. Start and finish times are expressed in the format DD/SS+HH:MM where DD is the day number, SS is the shift number, and HH:MM gives the hours and minutes into that shift. For this data set, assume that all schedules are built with respect to the beginning of shift one on work day one (1/1+00:00) and that time consists of a continuous sequence of work days, each consisting of two 7.5 hour shifts. Although reference to real dates and times would add realism to these problems, I did not want to introduce any constraints that might unnecessarily limit participation. The subjects of interruption, preemption, calendars, and realistic timescales may be the subject of a later series of problems.

Section 4 of this data set contains miscellaneous data required for the twelve problems that follow. The format and proper interpretation should be easy to infer from the problem statements and inspection of the data.

Section 5 of this data set contains C++ source code which implements routines that read and write the data found in sections 1, 2, and 3. It is provided courtesy of Mark Ringer. We cannot provide any specific support for integration of this code with a participants application program but we will be happy to respond to simple questions submitted to the e-mail address provided below.

Whenever zone constraints apply you can assume that the maximum occupancy in each zone is determined by the following table of zone occupancy limits.

Assembly Zone	Maximum Occupancy
Zone.Za	2
Zone.Zb	1
Zone.Zc	1
Zone.Zd	2
Zone.Ze	1
Zone.Zf	2
Zone.Zg	1
Zone.Zh	2
Zone.Zi	5
Zone.Zj	2
Zone.Zk	1
Zone.Zl	4
Zone.Zm	3

Whenever labor constraints apply you can assume that the allocations found in the following table of basic labor availability have been made for two 7.5 hour shifts each day. The specific start and finish times as well as break and meal times are immaterial in this first series of problems. Only the net duration of a shift is significant.

Labor Pool	Number Available in Shift 1	Number Available in Shift 2
Labor.Px	2	1
Labor.Py	3	1
Labor.Pz	2	2
Labor.Pw	3	2

The following problems are all based upon this single data set. However, new problems may be added at a later time that add new data or that impose restrictive interpretations on the existing data. Alternatively, new data that is consistent with this format and interpretation may be created at a later time to serve as the focus of the same set of problems. Please do not build any simplifying assumptions into your benchmarking software. For example, do not assume that the activities occur in precedence order and do not assume that step numbers are consistent with precedence order. Review the complete series of current problems and the general guidelines for submission of results before beginning work so that you do not make any design decisions that preclude solution of later problems.

Submission of Results

All participants are expected to provide a minimum amount of information with each submission so that their solutions can be quickly compared. It is expected that more detailed information can be obtained via e-mail or http access. We simply do not have time to validate all solutions submitted. Instead, all participants are expected to make their solutions available to others via e-mail or anonymous ftp whenever requested. The format for schedule data defined above should be adopted by all participants. Be prepared to respond to such inquiries before submitting your results. New, significantly better results, will naturally generate quite a bit of interest. Since the ability to import an existing schedule is required for the completion of Problem 5, all participants that reach this level will be in a position to import schedules created by others! The minimum information requested from each participant is:

1	Name of the individual making the submission plus the names of all team members.
2	Affiliation
3	E-mail address
4	Http address (optional) for your WWW home page where others can go to find out more about you, your organization, your methods, and your solutions.
5	Data set name, series number, problem number and metric value.
6	Wall clock time required to produce the solution
7	Hardware used
8	Solution method (critical path method, integer linear programming, simulation, constraint satisfaction, interative refinement/repair, etc.) Be prepared to provide more information upon request.

All submissions should be mailed to "benchmrx@neosoft.com". We will send acknowledgement of your submission after we have archived and posted your results. This should normally be accomplished within one week.

Do not attempt to solve all problems before submitting your first. Submit each problem result separately upon completion. Upon receipt we will append your submission to a cumulative list of submissions that will be accessible from the Planning & Scheduling Benchmarks WWW home page or via an e-mail request. These will be published in a simple spreadsheet format. When a sufficient number of submissions have been received we will create some summary reports and graphs. At a later time, we will validate and publish the schedules for the best solutions submitted. You are encouraged to set up a WWW home page that others can visit to learn more about you, your organization, your research, and your results. When a sufficient number are available, we will create a top-level document that organizes these pages according to methodology and other criteria. Please carefully inspect the data published for your submissions and immediately notify us of any errors using the e-mail address given above.

All participants are **strongly** encouraged to write the routines necessary to read and validate schedules so that they can perform independent validation of their own solutions before submission.

In the following problems, the objective is to produce schedules that satisfy all of the given constraints with no violations of precedence or resource availability. Time permitting, I will publish another data set that can be used for the same series of problems. I also hope to publish another data set that explores other technical issues including interruption and preemption, placement preferences, resource modeling, state constraints, etc. Suggestions for additional problems that can be based upon this first data set are welcome. Address your recommendations to "benchmrx@neosoft.com".

Whenever durations are to be reported, report your results in minutes. Whenever start or finish times are to be reported, report your results in the format DD/SS+HH:MM where DD is the day number, SS is the shift number, and HH:MM gives the hours and minutes into that shift. Work begins Day 1, Shift 1, 0 Hours, 0 Minutes.

Finally, I should make just a few comments on terminology. Several problems require that you "determine the minimal cycle time" for completion of the given work. I have not asked you to do the impossible. Some of the problems are NP-Hard and some are not. Just do the best that you can on each problem and report your result. **Problems 5 through 11 are all concerned with schedule update and revision. These problems are all based upon the schedule that I have provided in section 3 of the data set. Do not use a schedule that you have produced for these exercises. Doing so will make your results incomparable to others**. Some problems require that you reschedule some specific steps. In effect, this requires that they be removed from the current schedule and then placed on the schedule again subject to certain constraints specified in the problem. Problems 7 through 10 require very careful reading because the schedule revisions are intended to have very specific scope. Consult the questions submitted by participants before proceeding with these problems. Good Luck!

Problem Series 1

Problem 1: Precedence Scheduling

Ignore all labor and zone constraints, but respect all precedence constraints and determine the minimum cycle time for the completion of this assembly. Report the finish time of the last activity.

Problem 2: Resource Constrained Scheduling, Level Availability

Ignore all labor constraints, but respect all precedence and zone constraints and determine the minimum cycle time for the completion of this assembly. Never allow the zone occupancy to exceed the limits prescribed above. Report the finish time of the last activity.

Problem 3: Resource Constrained Scheduling, Non-Interruptible

Respect all precedence, zone, and labor constraints as described above and determine the minimum cycle time for the completion of this assembly. Never allow the zone occupancy to exceed the limits prescribed above; never allow the labor requirements to exceed the limits prescribed above; perform each activity from start to finish without stopping and guarantee that each activity finishes within the same shift as started. Report the finish time of the last activity.

Problem 4: Resource Constrained Scheduling, Interruptible

Respect all precedence, zone, and labor constraints as described above and determine the minimum cycle time for the completion of this assembly. Never allow the zone occupancy to exceed the limits prescribed above; never allow the labor requirements to exceed the limits prescribed above; perform each activity from start to finish stopping only between the first shift and the second shift or between the second shift and the first shift (interruption); and do not suspend any activities for either part or all of a shift (preemption). Report the finish time of the last activity.

Problem 5: Insertion of New Work

Now suppose that some new work, which was not part of the original assembly plan, must be added to the current schedule (found in section 3 of the data set). Add the following new activities (given in the format described above) to the schedule given in section 3 of the data set. In doing so, do not perturb the existing schedule in any way, but start and finish the new work as soon as possible while respecting the resource and precedence constraints given below. Report the start and finish time for just the new work.

NOTE: Problems 5 through 11 are all concerned with schedule update and revision. These problems are all based upon the schedule that I have provided in section 3 of the data set. Do not use a schedule that you have produced for these exercises. Doing so will make your results incomparable to others.

Step	Duration	P x	P y	P z	P w	Z a	Z b	Z c	Z d	Z e	Z f	Z g	Z h	Z i	Z j	Z k	Z l	Z m
new.step_001	03:30			2		1								1				
new.step_002	01:00			1						1								
new.step_003	02:30	1			1								1					
new.step_004	01:30		1					1										
new.step_005	00:30				2		1					1						
new.step_006	01:15			1	1							1		1				
new.step_007	03:05	1															1	
new.step_008	06:25			1			1											
new.step_009	00:10				1			1										
new.step_010	03:20		1										1					

Predecessor	Successor
asm_1.step_394	new.step_001
new.step_001	new.step_002
new.step_001	new.step_003
new.step_002	new.step_006
new.step_003	new.step_004
new.step_003	new.step_005
new.step_004	new.step_007
new.step_004	new.step_009
new.step_005	new.step_007
new.step_006	new.step_008
new.step_007	new.step_008
new.step_008	new.step_010
new.step_009	new.step_010
new.step_010	asm_1.step_518

Problem 6: Insertion of New Work with Delayed Start

Repeat Problem 5 but start and finish the new work as late as possible.

Problem 7: Daily Updates

Amend the schedule given in section 3 of the data set according to the following progress report for work completed through the second shift of the first day. Reschedule all work forward from day two shift one while respecting completed work, work in progress, planned starts, and delayed starts as annotated below. Report the finish time of the last activity.

Step	Status
asm_1.step_393	completed as scheduled
asm_1.step_366	completed as scheduled
asm_1.step_520	completed as scheduled
asm_1.step_228	delayed until after 3/1+00:00
asm_1.step_025	completed as scheduled
asm_1.step_383	completed as scheduled
asm_1.step_253	delayed until after 3/1+00:00
asm_1.step_378	completed as scheduled
asm_1.step_386	completed as scheduled
asm_1.step_480	not started, must be rescheduled
asm_1.step_089	completed as scheduled
asm_1.step_205	not started, must be rescheduled
asm_1.step_454	not started, must be rescheduled
asm_1.step_340	started at 1/2+05:00, will run normal duration
asm_1.step_430	started as scheduled, expected to finish at 2/1+03:30
asm_1.step_314	not started, must be rescheduled
asm_1.step_139	not started, must be rescheduled
asm_1.step_548	started as scheduled, expected to finish at 2/2+00:00
asm_1.step_048	planned to start at 2/1+00:00
asm_1.step_109	planned to start at 2/2+00:00
asm_1.step_368	delayed until after 3/1+00:00
asm_1.step_390	delayed until after 3/1+00:00

Problem 8: Daily Updates with a One-Week Horizon

Repeat problem 7 but restrict the schedule revisions to the first work week (days 1 through 5). Here the instructions must be specific and carefully worded in order to guarantee uniform interpretation. Because all work that cannot be completed by the end of the 5th day becomes a candidate for weekend overtime, reschedule all of the work that starts during the first week (including any steps that start in day 5 but finish on day 6). Do not allow any of the rescheduled work to finish later than the end of the 5th day and do not perturb any work that starts after the 5th day. Minimize the amount of work that fails to be rescheduled. Report the number of minutes of work that cannot be fit within the first week.

Problem 9: Temporary Perturbation of Labor Availability

Amend the schedule given in section 3 of the data set under the assumption that during the second, third, and fourth work weeks (days 6 through 20) some of the labor will be assigned to a special project as shown in the following table of labor availability for those weeks. Assume that the following allocation of labor is available for those weeks only and that the standard allocation of labor is available thereafter. Reschedule all work that finishes after the first week (including any steps that start in day 5 but finish on day 6). Do not allow any of the rescheduled work to start before day 6. Report the finish time of the last activity

Labor Pool	Number Available in Shift 1	Number Available in Shift 2
Labor.Px	2	0
Labor.Py	3	0
Labor.Pz	2	1
Labor.Pw	3	1

Problem 10: Temporary Perturbation of Labor Availability with 4 Week Horizon

Repeat Problem 9 but restrict the schedule revisions to only the second through fifth weeks (days 6 through 25). Here again the instructions must be specific and carefully worded in order to guarantee uniform interpretation. Because all work scheduled to be performed in the second through fourth weeks is affected by this change in labor availability, reschedule all work that starts or finishes in those weeks (including any steps that start in day 5 and finish in day 6 and any that start in day 20 and finish in day 21). Because we are trying to minimize the impact of this change, do not allow any of the rescheduled work to start before day 6 nor finish after day 25. Do not perturb any of the work already scheduled to start in the fifth week. Minimize the amount of work that fails to be rescheduled. Report the number of minutes of work that cannot be rescheduled.

Problem 11: Introduction of the Next Assembly

Assume that the schedule given in section 3 of the data set accurately represents the state of affairs at the beginning of day 16. Start production of a second assembly on day 16 shift one. Determine the minimum cycle time for the completion of the second assembly. Any of the work remaining on the first assembly can be rescheduled, but do not perturb any of the schedule prior to day 16 (since it is in the past) and do not exceed the current completion time for that assembly (since we have a customer waiting for its delivery). Report the finish time of the last activity of the second assembly.

Problem 12: Multiflow Production with Multi-Objective Optimization

In order to determine whether the allocation of labor shown in the following table of proposed labor availability is sufficient to for continuous production, make 8 copies of the activity data and rename the activity and zone constraints to create data for 8 assemblies (roughly six months of production). Schedule the start of a new assembly every 15 working days (day 1, day 16, day 31, day 46, etc.). Simultaneously minimize the cycle time for all assemblies while respecting the constraints prescribed in Problem 4. The goal is for all assemblies to have approximately equal (and minimized) cycle times but it is reasonable for the first and last assembly to have noticeably different cycle times from the others. Report the finish time for the last step of each assembly. Remember that the labor comes from four common pools but that work zones are specific to each assembly.

Labor Pool	Number Available in Shift 1	Number Available in Shift 2
Labor.Px	5	2
Labor.Py	4	2
Labor.Pz	3	1
Labor.Pw	5	3

Submissions

We will quickly publish any questions submitted by participants that might clarify the interpretation of the published data sets and problems. Likewise, we will quickly publish any comments submitted by participants. Short comments will be published in full, long comments will be summarized or will be included in full by reference to http address. Results submitted by participants should follow the guidelines published within the respective data sets. Results will be published only in summary form but participants should be prepared to respond to specific inquiries about their work. Participants are encouraged to create a WWW home page for publication of longer comments, detailed results, technical papers, summaries of products and services, etc.)

	Name	Problem	Time HH:MM:SS	Hardware	Metric	Method
1	M. Ringer	1	00:01:01	Pentium 90	27/1+01:29	feasible interval
2	M. Ringer	2	00:01:27	Pentium 90	45/2+02:46	feasible interval
3	M. Ringer	3	00:03:41	Pentium 90	57/1+06:25	feasible interval
4	M. Ringer	4	00:02:44	Pentium 90	56/1+05:40	feasible interval
5	N. Agarwal	1	00:00:05	HP 9000	27/1+01:29	CPM
6	N. Agarwal	2	00:00:05	HP 9000	40/2+02:54	CPM
7	N. Agarwal	3	00:00:05	HP 9000	47/1+06:25	CPM
8	N. Agarwal	4	00:00:05	HP 9000	57/2+00:09	CPM
9	N. Agarwal	5	00:00:05	HP 9000	21/1+00:35 29/1+03:20	CPM
10						

Participants

Name	Affiliation	E-Mail Address	WWW Address
Mark Ringer	Honeywell SRC	mringer@src.honeywell.com	
Nitin Agarwal	SAS Institute	sasnza@unx.sas.com	http://www.sas.com/

Questions

None at present.

Comments

None at present.

Revisions

The published problems and data will not be revised unless we discover an error or source of ambiguity that cannot be resolved by publishing a simple clarification. However, the first 12 problems do not really cover all important issues in Resource Constrained Project Scheduling. As time permits we will create new problems that exercise additional capabilities that would be important in a full operational system.

Additions

None at present.