LOCAL CONSTRAINT BASED ANALYSIS IN RESOURCE CONSTRAINED SCHEDULING

by

LİNET ÖZDAMAR

B.S. in I.E., Boğaziçi University, 1982

M.S. in I.E., Boğaziçi University, 1988

Submitted to the Institute for Graduate Studies in Science and Engineering in partial fulfillment of the requirements for the degree of

Doctor

of

Philosophy

Boğaziçi University

1991

ACKNOWLEDGEMENTS

I would like to express my gratitude to my thesis supervisor Prof. Dr. Gündüz ULUSOY for his guidance during this study.

I would also like to express my sincere thanks to Prof. Dr. Murat DINÇMEN and Doç. Dr. İlhan OR for serving on my thesis committee.

I also thank my colleagues Mr. Oktay GÜNLÜK and Mr. Nezih SOYDAN for their efforts in the implementation of the procedure.

LOCAL CONSTRAINT BASED ANALYSIS IN RESOURCE CONSTRAINED SCHEDULING

Resource constrained scheduling problems are generally solved by optimization techniques which cannot accomodate large size problems with respect to computation time. Yet, fast and near optimal schedules are required in actual dynamic environments. A method which has a quick response to dynamic conditions is the usage of dispatching rules which are tested and evaluated in simulation studies or against static optimal solutions. The disadvantage of heuristics lies in their performance dependence on problem characteristics.

Constraint based analysis (CBA) approach employed here stands midway between the latter two methods. The performance criterion is imposed on the scheduling process as a constraint together with resource limitations. These constraints are considered as locally essential at each scheduling decision point and activities are sequenced in accordance with the constraints.

In this research, the local essential conditions are explained in detail and the static and quasi-dynamic algorithms for resource constrained scheduling in which they are incorporated are conveyed. Static solutions for the previously solved project scheduling test problems obtained employing local CBA are compared with well-known heuristics as well as with the optimal solutions. Furthermore, CBA results for static job-shop problems representing various shop characteristics are compared with heuristics.

KAYNAK KISITLI ÇIZELGELEMEDE GEREKLİ YEREL KOŞULLAR

Kaynak kısıtlı çizelgeleme problemi genel olarak eniyi çizelgeleme teknikleri ile çözülür. Ancak bu yöntemler büyük problemlerde çözüm zamanı açısından yetersiz kalmaktadır. öte yandan gerçek dinamik ortamlarda eniyiye yakın ve hızla elde edilebilen çizelgeler gereklidir. Dinamik öğelere yanıt verilmesinin bir yolu benzetim çalışmalarında veya durağan olarak eniyi çözümlerle karşılaştırılıp değerlendirilen sezgisel yöntemlerdir. Ancak bu yöntemler düzenli olarak iyi çizelgeler yaratamamakta problem özelliklerine göre farklılık göstermektedir.

Bu çalışmadaki yaklaşım ise sözü geçen iki yöntemin arasında yer almaktadır. Performans kriteri çizelgeleme işlemi üzerinde bir kısıt olarak görülmekte ve kaynak kısıtları ile birlikte düşünülmektedir. Bu kısıtlar işlerin birbirlerine göre sıralarını her lokal karar verme anında belirlemektedir.

Bu çalışmada, zorunlu yerel koşullar ayrıntılı olarak anlatılmakta ve kaynak kısıtlı çizelgeleme için bu koşulları içeren durağan ve dinamik algoritmalar verilmektedir. Daha önce çözülmüş olan durağan proje çizelgeleme problemleri için yerel kısıt analizi yöntemi ile elde edilen sonuçlar hem eniyi çözümlerle hem de tanınmış sezgisel yöntemlerle karşılaştırılmıştır. Ayrıca, çeşitli atelye özelliklerini içeren durağan atelye problemleri için de kısıt analizi sonuçları ile sezgisel yöntemlerin sonuçları karşılaştırılmıştır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION

The subject of concern in this research is the development and implementation of local essential conditions with the aim of obtaining near-optimal time efficient solutions to the resource-constrained project and job-shop scheduling problems. The project scheduling problem indicated here is of the singly constrained, renewable resource type. Preemption of activities is not allowed and the objective is the minimization of makespan, C_{max}. The class of job-shop considered in this study is of serial or nonserial, machine constrained type where the jobs go through a known sequence of machines. The jobs may have nonzero release times. Parallel identical machine groups are assumed when the available resouce limit for certain machine types exceed unity. Precedence among jobs is also allowed and the objective considered is maximum tardiness, T_{max}. Total Work Content(TWK) based due date assignment is assumed for T_{max} performance evaluation.

The idea beneath the fore-mentioned essential conditions is to capture the above problems' intrinsic nature which heuristic rules fail to grasp. Furthermore the conditions developed in the following sections lead to local decisions and do not consider the scheduling problem at hand as a whole. This causes the implementation of the conditions to be quite efficient and much of the computational burden is eliminated.The latter enables the employment of essential conditions quasi-dynamically, i.e., in an event-controlled manner. The static

problem has to be resolved each time a disturbing event occurs during the progress of the project or work shop. Therefore dealing with the dynamic environment is through solving the static problem efficiently with respect to CPU times and quality of schedules in terms of optimum solution proximity.

The concept of essential or "essential" conditions is based on a process called "constraint-based analysis" [1,2,3] used in bringing out information employable as a decision aid in real-time resource-constrained scheduling. Constraint-based analysis(CBA) is performed in the latter context so as to provide sets of feasible schedules for the decision maker by iteratively finding precedence relationships necessary for feasibility with regards to due dates, release times and technological constraints and updating the time span during which an activity is free to be scheduled. CBA process is terminated either when it cannot find a complete schedule or when no feasible schedule exists with the available due dates or when it succeeds to find a complete sequence [1,4].

In the process of finding precedence relationships CBA considers all jobs to be scheduled on a machine, thereby viewing the problem globally. In consequence, CBA seems to have a computational handicap as stated in [2,4] and recommended to be employed in the static part of the scheduling module.

In this research the essential feasibility rules found in CBA have been developed with a different perspective and in such a manner that

they are valid for all possible instances at a decision point. Consequently, CBA has been localized with the expectation of near-optimal results and less computation time.

The research aims to contribute both to the general understanding of CBA and to heuristic procedures used in resource-constrained project and job-shop scheduling. The CBA project scheduling results are compared with the optimal solutions of previously tested problems [5,6]. The results for job-shop scheduling problems are compared with the well-known heuristic rules found in the literature.

II. LITERATURE SURVEY

The combinatorial aspect of the resource-constrained project scheduling problem has caused optimization approaches [7,8,9,10,11,12] to be computationally intractable for real-world problems.

Early heuristic approaches have been provided by Wiest [13] and by Thesen[14]. More recent heuristic researches [15,16] incorporate characteristics of the project and evaluate priority rules under multicriteria objectives.

Surveys and comparisons of heuristics to optimal solutions are given in Davis and Patterson [17], Herroelen [18] and Willis [19].There is general agreement on best heuristics to be used in project scheduling although none is proved to provide good solutions consistently [20].

It should be noted that the Resource Scheduling Method(RSM) heuristic given in [17,20] compares all pairs of schedulable activities and decides upon precedence among a pair of activities according to the project duration increase caused by the precedence of one activity over the other. The rule has thus some conceptual similarity to CBA approach.

The heuristics which have been found as best in project scheduling are picked up and compared with optimal solutions so that CBA performance is considered fairly against a background of well reputed heuristic performance.

As for the job-shop problem, simulations have been carried out by several authors [21,22,23] in order to determine best heuristics with respect to tardiness, flow time and other criteria. No dispatching rule

seems to be consistently the best, but there are generally accepted good rules for exogenously determined due dates and internally determined due dates respectively as given in a survey by Blackstone et al.[24]. Baker and Kanet [25] also propose a new rule for the tardiness criterion. These dispatching rules are compared with CBA solutions for the job-shop problems with respect to T_{max}.

Recent trends in project management and resource-constrained scheduling consist of bringing a dynamic aspect to the problems in a disturbed environment. Two generally accepted disturbances are changes in resource availabilities and the arrival of new jobs. Muhlemann et al.[26] experiment on the performance of priority rules coupled with the frequency of rescheduling. Raman et al. [27] tranform an automated environment scheduling problem into resource-constrained project scheduling problem and apply static solution procedures on a rolling horizon basis.The effects of schedule revisions are given in Yamamoto and Nof [28] and improvement of criteria due to rescheduling is discussed. Norbis and Smith [29,30] provide both static and event-controlled(quasi-dynamic) versions of a two level heuristic for the resource-constrained scheduling problem. Another quasi-on-line approach to scheduling is given by Slomp et al. [31] for flexible manufacturing systems. Dumond and Mabert [32] evaluate priority and due date assignment rules according to several criteria in a dynamic environment where projects arrive randomly over time.

All the above real-time scheduling approaches consider the dynamic scheduling problem as a series of static project(job shop) scheduling problems, updating the system each time a disturbance occurs. In consequence,the problem is to find a time-efficient and high quality(in terms of proximity to optimal solutions) procedure for the static problem.

III. LOCAL ESSENTIAL CONDITIONS IN PROJECT SCHEDULING

As previously mentioned in the introduction the development of essential conditions is based on the concepts introduced by Erschler et al.[1,2]. The authors have developed essential conditions for two major instances: Pairwise conflict and all pairs in conflict cases which are discussed in detail as Cases A1 and A2 in the following sections. Their approach is different from the one adopted here in the following sense: The authors propose to use their essential conditions in order to find sets of feasible schedules compatible with resource and time(duedate) constraints. They consider all jobs in the workshop simultaneously and the two cases involved concern only "permanent" conflicts(conflicts which always exist with respect to daily resource limits). Their procedure does not take temporary conflicts(conflicts which exist only at a certain decision point with respect to current resource levels) into account. Consequently, when their essential conditions are not tight enough to produce a complete schedule their feasible schedules become partial schedules based on these two cases. The partial schedules are fed into a decision aid module linked to an on-line workshop monitoring system to be employed by schedulers. The partial schedule generation is claimed to be computationally burdensome as it involves the identification of all pairwise conflicts among overlapping jobs. Additionally, partial schedule generator is stopped when no feasible schedules can be generated with the current due dates.

In this study, however, there exists an infeasibility check before the corresponding essential conditions are applied. The reason is to identify infeasibility at an early stage of the procedure and avoid imposing unnecessary precedence relationships on the activities. Furthermore, the utilization of local CBA enables the scheduler to obtain complete schedules without the requirement of a decision aid module. At the times when the local essential conditions are not sufficiently strong to identify precedence relationships the procedure decides according to a predefined dispatching rule.

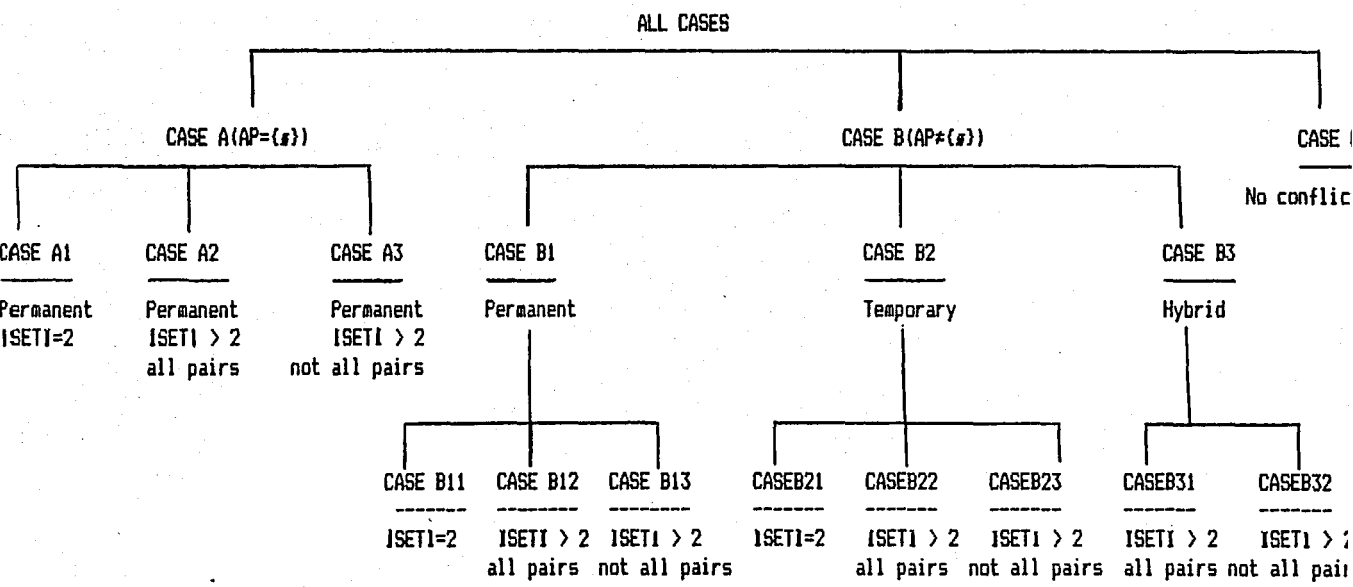
The essential conditions developed here are used in both project and job-shop scheduling contexts. The procedure can be used for both C_{max} and T_{max} objectives.

Developing such local conditions requires the consideration of all possible instances that can occur at any scheduling point. What is meant by instances is made clear by the following:

An instance might involve activities in progress which leads to the concept of temporary conflicts. Furthermore, there might be instances when pairs of activities are not in any type of conflict but when considered as a group cannot be scheduled concurrently due to resource limitations.

A summary of all possible instances with their respective brief descriptions is found in Table 3.1. Notation included in Table 3.1. and in other sections is given in Appendix.

TABLE 3.1. Summary of all possible cases.



3.1. Definitions of Possible Cases and Essential Conditions for Each Case

CASE A (AP={ø})

No activities are in progress at t_{now}.

Case A

Definition:

Permanent conflict: Permanent conflict exists among two activities when

$$\sum_{k=i,j \in SET} r_{k,h} > R_h \quad \text{for one or more resource types h.}$$

Case Description:

Permanent conflict exists among pairs of activities.

Case A1 (|SET|=2)

Only two activities are in permanent conflict. (SET = {i, j})

Checking infeasibility :

If $(lft_{max} - t_{now} < \sum_{k=i,j} p_k)$, then project duration must be increased by the following amount :

α = project duration increase = $t_{now} - lft_{max} + \sum_{k \in SET} p_k$

Proof:

A: Suppose i has lft_{max} over i,j and $\sum_{k=i,j} p_k > lft_{max} - t_{now}$.

This implies:

$p_i + p_j > lft_i - t_{now}$

$p_j > lft_i - p_i - t_{now}$

$p_j + t_{now} > lst_i$

$eft_j > lst_i \Rightarrow j$ cannot « i.

$\Rightarrow$ i « j. $\Rightarrow \sum_{k=i,j} p_k + t_{now} < lft_j$ which is contradictory to A.

For all $j \in U$, lst_k, lft_k are updated by α, i.e. $lst_k = lst_k + \alpha$; the same goes for lft_k.

The final project duration increase is the maximum α over all pairwise comparisons.

Since α is the consequence of conflict infeasibility checks only and since CPM is solved only once at the beginning of the procedure it is possible that at any time during the procedure $lft_i < t_{now}+p_i$, then $\alpha = t_{now}+p_i-lft_i$ and lft_i and lst_i are updated accordingly.

Precedence Rule:

a. If $lst_i < eft_j$, then i precedes j. $\{i,j \in SET\}$

b. If $lst_j < eft_i$, then j precedes i.

Proof (a):

Suppose $lsti<eft_j$ and $j \ll i$.

$C_j=eft_j > lft_i-p_i$

Since $j \ll i$, $C_i=C_j + p_i > lft_i$ which violates feasibility of current C_i. ∎

Both conditions (a) and (b) cannot occur simultaneously because of the infeasibility check undergone in the first place.

Proof:

Suppose a) $lst_i < eft_j$ and b) $lst_j < eft_i$.

By infeasibility check,

if j has lft_{max}: $lft_j-t_{now} \geq p_i+p_j$

$\Rightarrow lst_j-t_{now} \geq p_i$

$\Rightarrow lst_j \geq eft_i$ which contradicts (b).

Similarly, if i has lft_{max}, $lst_i \geq eft_j$ which contradicts (a). ∎

If none of the conditions are true, then the sequence is arbitrary with respect to CBA and activities are scheduled according to the previously selected dispatching rules.

Example A1

SET = {A,B} R_1= 5 R_2= 6

Act.	Res.Req.(1)	Res. Req.(2)	p	est	eft	lst	lft
A	3	3	3	3	6	4	7
B	4	3	3	3	6	6	9

$\alpha = 0$

A ≪ B .

Case A2 (|SET| > 2)

Case description:

More than two activites are in conflict and all possible pairs are in permanent conflict.

Checking infeasibility is done in pairwise fashion as in case A1, and also groupwise, i.e.,

$$\alpha = \max\left\{ \max_{\text{all pairs}} \{0, t_{now} - lft_{max} + \sum_{k=i,j} p_k\}, \max \{0, t_{now} - \max_{i \in SET} \{lft_i\} + \sum_{k \in SET} p_k\} \right\}$$

Precedence Rule:

Step 1. Check pairwise precedence as in case A1. If no relationships are found or if pairwise comparisons do not diminish the size of SET' to unity, go to Step 2.

Step 2. Check group precedence for every $i \in SET'$:

a. If $[\max_{j \in \{SET'-i\}} \{ lft_j \} - t_{now}] < \sum_{k \in SET'} p_k$, then V $j \in \{SET'-i\}$ precedes i.

where V is the boolean OR.

b. If $[lft_i - t_{now}] < \sum_{k \in SET'} p_k$, then V i precedes $j \in \{SET'-i\}$.

Proof(a):

Suppose $[\max_{j \in \{SET'-i\}} \{ lft_j \} - t_{now}] < \sum_{k \in SET'} p_k$ and V i precedes $j \in \{SET'-i\}$.

$\Rightarrow \max_{j \in \{SET'-i\}} \{lft_j\} < \sum_{k \in \{SET'-i\}} p_k + p_i + t_{now}$

$\Rightarrow \max_{j \in \{SET'-i\}} \{lft_j\} < \sum_{k \in \{SET'-i\}} p_k + eft_i$

$\Rightarrow \max_{j \in \{SET'-i\}} \{lft_j\} - \sum_{k \in \{SET'-i\}} p_k < eft_i$

$\Rightarrow lst_{<SET'-i>} < eft_i$ which violates the feasibility of current $C_{SET'-i}$

Proof(b):

Suppose $[lft_i - t_{now}] < \sum_{k \in SET'} p_k$ and V $j \in \{SET'-i\}$ precedes i

$\Rightarrow lft_i - p_i - t_{now} < \sum_{k \in SET'-i} p_k$

$\Rightarrow lst_i < \sum_{k \in SET'-i} p_k + t_{now}$

$\Rightarrow lst_i < eft_{SET'-i}$ which proves that i has to precede at least one activity $j \in \{SET'-i\}$ for current C_i feasibility.

For each $i \in SET'$ do comparisons 2a and 2b. Note that both conditions cannot occur for an activity i.

Proof:

$$\left.\begin{array}{l}\left[\max_{j\in(SET'-i)}\{lft_j\}-t_{now}\right] < \sum_{k\in SET'} p_k \\ \left[lft_i - t_{now}\right] < \sum_{k\in SET'} p_k\end{array}\right\} \Rightarrow \max_{j\in SET'}\{lft_j\}-t_{now} < \sum_{k\in SET'} p_k$$

which violates the infeasibility check.

If none of 2a or 2b occur, no relationship exists between i and (SET'-i) and dispatching rules are used.

Example A2

SET = (A,B,C) R_1= 4 R_2= 3

Act.	Res.Req.(1)	Res.Req.(2)	p	est	eft	lst	lft
A	2	2	3	5	8	9	12
B	4	1	2	5	7	9	11
C	3	2	4	5	9	11	15

Step 1. No relationships are found.

Step 2.

i=A A ≺ B or A ≺ C

i=B B ≺ A or B ≺ C

i=C A ≺ C or B ≺ C

Result: SET"=SET - (C). Calculate priorities for A and B.

Case A3 (|SET|>2)

Case description:

A group of activities is in conflict and every pair of activities is not in conflict. This case includes the instance of no pairwise conflicts.

Definition:

S: Minimal conflict set; A set such that if any member i is dropped from S, then {S-i} can be processed simultaneously. ($S \subset SET'$)

Precedence Rule:

Step 1. Check pairwise infeasibility and precedence as case A1. Pairwise checks will diminish the size of SET' since the ultimate successors are deleted from SET.(SET -> SET')

Step 2. If SET' is not nil, check group precedence for every $S \subset SET'$ and $i \in S$:

a. If $lst_i < \underset{j \in \{S-i\}}{eft_j}$, then i precedes { S-i }.

b. If $lst_j < \underset{j \in \{S-i\}}{eft_i}$, then { S-i } precede i.

Proof(a):

Suppose $lst_i < \underset{j \in \{S-i\}}{eft_j}$ and { S-i } $\not\ll$ i.

By definition {S-i} is a nonconflict set. Furthermore, no $j \in \{S-i\}$ is in pairwise conflict with i since pairwise conflicts are dealt with in Step 1 and S consists of more than two activities. Consequently, {S-i} can be considered as a single activity in pairwise conflict with i. This implies:

$$eft_{<S-i>} = \min_{j \in \{S-i\}} \{eft_j\}$$

since the completion of any $j \in \{S-i\}$ provides the possibility of concurrent scheduling of i and {S-i}. Then,

$$lst_i < \underset{j \in \{S-i\}}{eft_j} \Rightarrow lst_i < \min_{j \in \{S-i\}} \{eft_j\}$$

which violates the current value of C_i.

Note that in this case both conditions can be true since groupwise infeasibility check which is quite involved is not applied here. If no consistent sequencing can be obtained or if SET" still consists of conflict groups using the above, then dispatching rules are used to decide upon the activities in SET".

Example A3

SET={A,B,C,D} R_1= 5 R_2= 7

Act.	Res.Req.(1)	Res.Req.(2)	p	est	eft	lst	lft
A	2	3	4	5	9	8	12
B	1	3	3	5	8	8	11
C	3	2	4	5	9	7	11
D	1	2	6	5	11	7	13

Step 1. No pair conflicts exist.

Step 2. S = {A,B,C} i=A 2a. X 2b. Yes. {B,C} « A.

i=B 2a. Yes. B « {A,C} 2b. X

i=C 2a. Yes. 2b. Yes. => No relation.

S = {A,C,D} i=A 2a. Yes. 2b. Yes. => No relation.

i=B 2a. Yes. 2b. Yes. => No relation.

i=C 2a. Yes. 2b. Yes. => No relation.

S: {A,B,D} i=A 2a. X 2b. Yes. {B,D} « A.

i=B 2a. Yes.B « {A,D} 2b. X

i=D 2a. Yes. 2b. Yes. => No relation.

Result: SET"= SET' - {A}. Schedule B,C,D at t=5.

Case B(AP≠{∅})

Case Description:

Activities are in progress at t_{now}. (Note that for all B subcases if $tconf_j \geq lft_j$, $j \in SET$, then case A rules are applied.)

Case B2

Case Description:

Any conflict that may exist between activities is temporary.

Definition:

Temporary conflict: A pair of competing activities are in conflict with respect to the available resource levels at t_{now} but not to the daily resource limit, R_h, i.e.

$$R'_h \leq \sum_{k=i,j} r_{k,h} \leq R_h.$$

$t^{conf}_{i,j}$:The earliest time when both i and j can be processed simultaneously using the available resources and the ones provided by the completion of activities in progress.

Case B21(|SET|=2)

Case Description:

Two activities are in temporary conflict,i.e. SET = {i,j}.

Checking infeasibility:

$$\alpha = \min \left\{ \min_{i \in SET} \left\{ \max_{j \in SET} \left\{ 0,\ t^{conf}_{i,j} + p_i - lft_i \right\} \right\},\ \max \left\{ 0,\ \sum_{k \in SET} p_k + t_{now} - lft_{max} \right\} \right\}$$

Proof:

Suppose $\sum_{k=i,j} p_k > lft_i - t_{now}$ (where i has lft_{max}) and $lst_i \geq t^{conf}_{i,j}$.

Then, $lst_i < eft_j$ and $lst_j < eft_i$ but i can be scheduled at $t^{conf}_{i,j}$ when i and j can be run concurrently. Since by the first part of the infeasibility check:

$$t^{conf}_{i,j} + p_i \leq lft_i$$

current C_i is not violated in spite of the second part of the infeasibility check.

Similarly, when the first part of the infeasibility check is violated and the second part holds current value of C_{max} is preserved.■

Precedence Rule:

a. If $lst_i < t^{conf}_{i,j}$ and $lst_i < eft_j$, then i precedes j.

b. If $lst_j < t^{conf}_{i,j}$ and $lst_j < eft_i$, then j precedes i.

Proof(a):

Suppose $lst_i < eft_j$, $lst_i < t^{conf}_{i,j}$ and $j \ll i$

Then,

$lst_i < C_j \Rightarrow lft_i < C_j + p_i$.

$lst_i < t^{conf}_{i,j} \Rightarrow lft_i < t^{conf}_{i,j} + p_i$.

$\Rightarrow$Current C_i is violated.■

Both conditions cannot occur because of the infeasibility check. If none of 2a or 2b occur, then no sequencing can be established and dispatching rules are used.

Example B21

SET = (A,B) R_1= 7 R'_1= 5 R_2=5 R_2'=3

Act.	Res.Req.(1)	Res.Req.(2)	p	est	eft	lst	lft	Resources released: 2 units(1) at t=3
A	4	2	3	0	3	3	6	1 unit(2) at t=2
B	3	2	4	0	4	2	6	$t^{conf}_{A,B}$ =3

α=0

Result: B « A

Case B22(|SET|>2)

Case Description:

More than two schedulable activities are in conflict and every pair of activities are in temporary conflict.

Precedence Rule:

Step 1. Pairwise infeasibility and precedence checks are applied as in Case B21.If no relationships are found or if the size of SET' is not unity after pairwise comparisons dispatching rules are used.

Case B23(|SET| > 2)

Case Description:

A group of activities are in conflict and every pair is not in conflict. The conflicts among pairs of activities are temporary ones.

Definition:

(Note: S is defined according to R'_h.)

AC_s: Subset of activities in A which when completed enable the earliest simultaneous progress of S, s.t.

$$\sum_{j \in AC_s} r_{jh} + R'_h \geq \sum_{i \in S} r_{ih}$$

$ERT_{i,s}$: The earliest time when activity i can progress simultaneously with {S-i} using resources provided by the completion of an activity $j \in \{S-i\}$ or $j \in AC_s$.

$$ERT_{i,s} = \min \begin{Bmatrix} \min_{j \in \{S-i\}} \{eft_j\} \\ \max_{j \in AC_s} \{eft_j\} \end{Bmatrix}$$

Precedence Rules:

Step 1. Check pairwise infeasibility and precedence as in Case B21. The ultimate successors are deleted from SET.(SET -> SET').

Step 2. If SET'is not nil,check group precedence for all$\subset$S SET'and $i \in S$:

a. If $lst_i < ERT_{i,s}$ and $lst_i < \min_{j \in \{S-i\}} eft_j$,

then i precedes {S-i}.

b. If $lst_j < ERT_{j,s}$ and $lst_j < \min_{j \in \{S-i\}} eft_i$,

then {S-i} precede i.

Proof(a):

Suppose $lst_i < ERT_{i,s}$ and $lst_i < \min_{j \in \{S-i\}} eft_j$ and {S-i} $\ll$i.

As {S-i} is a noncoflict set, {S-i} and i can be regarded as a pair of activities in conflict with each other up to the time $ERT_{i,s}$ when they can be run concurrently. Then,

$$lst_i < \min_{j \in \{S-i\}} eft_j \Rightarrow lst_i < \min_{j \in \{S-i\}} \{eft_j\} \Rightarrow lst_i < t_{now} + \min_{j \in \{S-i\}} \{p_j\}$$

With the second condition: $lst_i < ERT_{i,s}$ the current C_i is violated.

If no consistent sequencing can be established or if SET" still consists of conflict groups dispatching rules are used to schedule j∈SET".

Example B23

SET = {A,B,C,D} R_1 = 10 R'_1= 7

Act.	Res.Req.(1)	p	est	eft	lst	lft	Resources released: 2 units at t=8
A	2	4	5	9	8	12	
B	4	2	5	7	7	9	
C	2	3	5	8	7	10	
D	2	4	5	9	7	11	

Step 1. No pair conflicts.

Step 2. S = {B,C,D} ERT_B=8 ERT_C=7 ERT_D=7

S = {A,B,C} ERT_A=7 ERT_B=8 ERT_C=7

S = {A,B,D} ERT_A=7 ERT_B=8 ERT_D=7

The sole lst which is less than its corresponding ERT is of activity B.

Therefore, check 2a. for i=B for all S:

S = {B,C,D} => B ≪ {C,D}

S = {A,B,C} => B ≪ {A,C}

S = {A,B,D} => B ≪ {C,D}

Result: Assign B at t=5. Select among A,C,D by using priority rules.

Case B1(AP≠{∅})

Case Description:

There exist activities in progress. The types of conflict among pairs of activities for all the following three cases are permanent, i.e. $\Sigma r_{ki} \geq R_i$. In these cases the resources provided by the activities in $k=i,j \in SET$ progress do not help .

Case B11 (|SET|=2)

Case Description:

Two activities are in permanent conflict. (Note that each activity is schedulable under the available resource level at t_{new}.)

Precedence Rule:

Rules of Case A1 are applied in this case.

Case B12(|SET|>2)

Case Description:

More than two activities are in conflict and each pair is a permanent conflict pair.

Precedence Rule:

Rules of Case A2 are applied.

Case B13 (|SET|>2)

A group of activities are in conflict and but each pair is not a conflict pair. The conflict pairs are permanent ones.

Step 1. Pairwise infeasibility and precedence checks are applied as in Case A1. (SET->SET')

Step 2. Group precedence rules are applied as in Case B23 where S is the minimal conflict set according to the available resource level, R'_h.

Case B3 (AP≠{ø},|SET|>2)

Case Description:

Hybrid cases: A hybrid case occurs when some of the activity pairs form permanent conflicts and some temporary conflicts. Hybrid cases occur when activities are in progress and there are more than two activities in SET.

Case B31

Case Description:

More than two activities are in conflict, but every pair is in either permanent or temporary conflict.

Precedence Rules:

Step 1. Apply infeasibility checks and precedence rules of Case A1 for permanent conflict pairs and rules of Case B21 for temporary ones. If no relationships are found or if the size of SET' is not reduced to unity, dispatching rules are used.

Case B32

Case Description:

Group of activities are in conflict, but not every pair is a conflict one. The conflict pairs are in either temporary or permanent conflict.

Precedence Rules:

Step 1. Pairwise infeasibility and precedence rules are applied as in Step 1 of Case B31. (SET->SET')

Step 2. Group precedence rules of Case B23 are employed.

Case C

Case Description:

No activites in conflict. (SET = { ϕ })

Assign i $\in$ { r } at t_{now}.

3.2. CBA Scheduling Algorithm for the Static Resource Constrained Scheduling Problem

The local essential conditions discussed in the previous section can be utilized through the following algorithm.

Step 1. Initialize: $t_{now}=0$, r={Activities with no predecessors}, AP={ϕ}, U={All activities}, UA={ϕ}, $R'_h=R_h$. Solve CPM.

Step 2. Determine and evaluate SET considering all resource types h and identify types of cases fitting SET.

Step 3. Assign activities as indicated in the detailed description of the cases.

Update: U={U-UA}

AP={AP+UA}

$$R'_h=R'_h-\sum_{j\in UA} r_{kh}$$

If U = ϕ, stop.

Step 4. Update: $t_{new} = \min_{j \in AP} \{C_j\}$

$R'_i = R'_i + \sum_{j \in FIN} r_{ji}$

Γ={Γ+activities schedulable with respect to R'_i and completed predecessors}

$\alpha = \max_{i \in \Gamma} \{0, t_{new}+pi-lft_i\}$; $lft_i = lft_i+\alpha$; $lst_i = lst_i+\alpha$ for $i \in \Gamma$.

AP={AP-FIN}

If $\Gamma = \emptyset$ then repeat Step 4.

Otherwise, go to Step 2.

3.3. Quasi-Dynamic Application of Local CBA

As mentioned previuosly, the efficiency of local CBA enables its utilization in a quasi-dynamic environment. The generally accepted types of disturbances in resource constrained scheduling are as follows:

1. Changes in the available resource levels.
2. Changes in activity durations, adding or deleting activities.

In all types of disturbances activities in progress at the disturbance time are not descheduled unless compulsory and the current schedule of activities assigned up to then is preserved. When an activity in progress is descheduled then preempt-repeat type of process is utilized. The procedure to be followed by the two types of events are given as follows:

Event type 1

1a. Increasing resource level h by δ_h at t_{evt}:

Update $t_{now}=t_{evt}$

$R'_h = R'_h + \delta_h$

Γ = {Γ+activities schedulable with respect to new R'_h}

If Γ is not nil, go to Step 2 of the scheduling algorithm.

Otherwise, go to Step 4.

1b. Decreasing resource level h by δ_h at t_{evt} by specifying which activity i* is interrupted:

Update $t_{now}=t_{evt}$

$R'_i = R'_h - \delta_h + r_{i*h}$

AP = {AP-i*}

U = {U+i*}

Γ = {Γ+activities schedulable with respect to new R'_h}

If Γ is not nil, go to Step 2 of the scheduling algorithm.

Otherwise, go to Step 4.

When changes in resource levels occur a duration for which the change is valid is also specified and the scheduling algorithm takes this duration into consideration while applying rules during the latter period of time.

Event type 2

2a. Changes in the durations of activities unassigned yet or in progress at t_{evt}:

Update $t_{now}=t_{evt}$

Solve the CPM for the remaining network denoting the durations of $j \in AP$ as: $p_j = Start.Time_j + p'_j - t_{now}$ where p'_j is the updated duration.

With the new est and lft values thus obtained go to Step 4 of the scheduling algorithm.

2b. Adding or deleting one or more activities unassigned yet at $t=t_{evt}$:

Update $t_{new} = t_{evt}$ Solve the CPM for the remaining network, update r.

If r is not nil, go to Step 2 of the scheduling algorithm.

Otherwise, go to Step 4.

IV. HEURISTIC RULES AND CBA OPTIONS IN PROJECT SCHEDULING

4.1. CBA Options Used in Local CBA Implementation

It is observed in the statement of the conditions that there might arise situations where the conditions lack the sufficiency to select the activity(ies) to be scheduled at some decision point. In such a case, the sequence of activities is arbitrary and the problem remains unsolved.[4] In order to go on with the procedure, one of the four dispatching rules are summoned to get rid off the arbitrariness. It should be emphasized that the dispatching rules are activated only after the failure of pairwise and groupwise precedence checks.

The above mentioned dispatching rules are MINSLACK(Minimum Slack Time), LFT(Minimum Latest Finish Time), WRUP(Weighted Resource Utilization Ratio and Precedence) and OC(Opportunity Cost) rules. LFT has been verified to outperform MINSLACK(Minimum Slack Time), RSM(Resource Scheduling Method), SIO (Shortest Imminent Operation) and GRD(Greatest Resource Demand) rules by previous researches [6,20]. WRUP is found by Ulusoy and Ozdamar [20] to be most effective among all the latter rules for specially designed problems representing all kinds of network/resource characteristics. On the other hand, OC which is introduced in this study has not been tested previously. OC is a rule based on conflict set logic and an indicator function consisting of network/resource characteristic values. The definitions of all four rules are found below:

MINSLACK: Priority is given to the activity with the minimum total slack time.

LFT: Priority is given to the activity with the minimum latest finish time.

WRUP: Priority is given to the activity with the minimum value, Z_i where:

$$Z_i = w_p * np_i + (1 - w_p) * \sum_h r_{ih}/R_h$$

Since project duration is affected by the selected precedence weight the best solution is obtained by increasing w_p with increments of 0.1 starting from the lowest limit of [0-1] interval. The size of the increment has been investigated previously [20] and the schedule duration has been found to be insensitive to increments of size less than 0.1.

OC: Priority is given to the set of activities with the minimum value, $OC_{s'}$. The definitions of variables related to this rule is given in Appendix. It should be noted that S' contains a single activity when ISETI=2 or when all pairs are in conflict.

When using OC rule, $produr_{s'}$ is calculated in all cases by excluding S' from the partial network of unassigned activities at t_{now} and updating the est_i of unassigned activities to the earliest possible starting time according to the resource availability based on the assumption that activities in S' have been scheduled at t_{now}. Forward

CPM is then applied to the latter network assuming unlimited resource availability to obtain $produr_{s'}$.

The second component of OC rule is the excess resource ratio which reflects the resource requirements of activities in excess of available resource levels assuming that all activities are scheduled at their updated earliest start times.

The opportunity cost($OC_{s'}$) of assigning S' considers both the project duration increase and the excess resource ratio in equal terms, i.e., they are given equal weights by the multiplication of the two terms. On the other hand, summing the two terms diminishes the weight of the excess resource ratio.

4.2. Dispatching Rules Used for Comparing Local CBA Results

Four dispatching rules are used as bases of comparison against CBA: MINSLACK, LFT, RSM and WRUP.These rules have been found to be the more successful by previous researches [6,17,20]. The reason for comparing heuristic rule performance against optimal results is to eliminate the problem specific evaluation of CBA performance.

V. TEST PROBLEM CHARACTERISTICS

The project scheduling problems for testing CBA and their optimal solutions have been made available by Christofides, Alvarez-Valdes and Tamarit [5] and by Alvarez-Valdes and Tamarit [6].In the former research 40 problems are solved to optimality by an efficient branch and bound procedure. In the latter study, 144 problems are solved by a variety of heuristic rules and most problems of small to medium size and a few large size problems are solved to optimality.

The first 40 problems have the following characteristics [5]: Loosely constrained problems have Utilization Factor(UF) values, which represent the ratio of total work content to available resource levels, in the interval [0.5-0.99] while tightly constrained ones have their UF values in the interval [1-1.5]. The mathematical expression of UF is found in the Appendix. Durations lie in the range of [1-9] and resource requirements in the range [0-6]. All problems consist of 25 activities and three resource types. Availability limit is 6 for all three types of resources. The complexity ratio,i.e., the ratio of the number activities to the number of events, is in the range [1-3]. The average solution time on a UNIVAC 1100 computer is 3.8 CPU sec.

The second batch of problems selected are medium to large size problems and have the following characteristics [6]: 36 problems have 51 activities and 16 others have 103 activities. All of the 51-activity and two of the 103-activity problems have their optimal solutions available.The remaining 14 problems of 103 activities have

their best solutions available instead of optimal solutions. Optimal solutions for these problems could not be found because of time considerations. Medium and large size problems have 6 resource types. Complexity ratio interval is [1-3] in 51-activity problems and [1.2-1.4] in 103-activity problems.Low complexity-large size problems have been chosen for testing in this research since they are difficult problems to solve and project durations as well as solution CPU times deteriorate for this type of problems. CBA is thus tested under most exerting circumstances. The resource constraints vary from relaxed to tight as in the first batch of problems. CPU times of the optimal solutions are not available whereas priority rules are claimed to take 0.1-0.2 sec. for 51-activity problems and 3 sec. for 103-activity problems on a BULL DPS8/49 computer.

VI. DISCUSSION OF PROJECT SCHEDULING RESULTS

Local CBA is observed to obtain solutions of about 80 per cent better than the heuristic rules with respect to optimal solution proximity. Table 6.1. summarizes the results obtained by CBA and the selected four heuristic rules. For each group of problems, i.e., 25-activity, 51-activity and 103-activity problem groups, the average percentage ratio of project durations over optimum solutions are stated with their respective standard deviations. CBA's performance with respect to the latter ratio is considerably better than the best of the four heuristics. CBA-Best results are on the average about 1.82 per cent above optimum solutions versus 3.44 per cent of Heuristic-Best. Furthermore, CBA finds optimal results 56.41 per cent of the time and overall best solution percentage is 92 per cent. On the other hand, the latter values are 34.61 per cent for obtaining optimal solutions and 55.12 per cent for obtaining best solutions in Heuristic-Best results. If on the other hand, the best of heuristic rules, i.e. LFT, is compared to the best of CBA options, i.e. CBA-WRUP, then it is observed that CBA-WRUP finds on the average 2.25 per cent above optimum results versus 4.64 per cent of LFT's. CBA-WRUP's optimal finding rate is 51.28 per cent versus 21.79 per cent of LFT's. It is also observed from Table 6.1.that overall best result percentage above optimum which is 1.54 per cent is quite near to CBA's value, 1.83 per cent. The same claim is true when 14 nonoptimal problems are included into the comparison basis.

TABLE 6.1. Summary of results for project scheduling problems.

							CBA			
RULES:	MINSLACK	LFT	RSM	WRUP	Heur-Best	CBA-Best	WRUP	LFT	MINSLACK	OC
PROBLEMS WITH 25 ACTIVITIES (40 problems)										
AVG ± σ	4.61±	4.68±	7.15±	5.40±	3.00±	1.37±	2.03±	3.62±	3.54±	3.93±
(%)	3.7	4.23	7.41	5.64	3.57	2.51	3.29	4.19	3.51	5.06
OPT	22.50	27.50	22.50	27.50	40.00	62.50	57.50	40.00	40.00	42.50
(%)										
GLOBAL BEST(%)	32.50	37.50	30.00	37.50	62.50	92.50	77.50	52.50	50.00	65.00
(0.95 ± 2.20 %)										
CBA-BEST	-	-	-	-	-	-	87.50	57.50	57.50	70.00
(%)										
PROBLEMS WITH 51 ACTIVITIES (36 problems)										
AVG ± σ	4.85±	4.72±	6.05±	5.36±	3.65±	2.39±	2.53±	3.91±	3.98±	3.35±
(%)	4.80	4.12	4.62	4.97	3.98	3.13	3.23	3.70	4.16	3.81
OPT	22.22	16.66	11.11	22.22	28.94	50.00	44.44	19.44	27.77	33.33
(%)										
GLOBAL BEST(%)	22.22	25.00	13.88	30.55	47.22	91.67	80.55	36.11	44.44	50.00
(2.27 ± 3.08 %)										
CBA-BEST	-	-	-	-	-	-	91.66	38.88	44.44	58.39
(%)										
PROBLEMS WITH 103 ACTIVITIES (16 problems)										
AVG ± σ	5.85±	5.12±	21.20±	10.05±	4.13±	1.68	2.00±	3.79±	4.63±	5.65±
(%)	3.00	3.35	8.94	6.46	3.11	1.96	2.14	2.81	2.74	4.28
GLOBAL BEST(%)	6.25	6.25	00.00	6.25	18.75	93.75	75.00	37.50	18.75	12.50
(1.73 ± 2.08 %)										
CBA-BEST	-	-	-	-	-	-	81.25	37.50	18.75	12.50
(%)										

TABLE 6.1. Summary of results for project scheduling problems (continued).

							CBA			
RULES:	MINSLACK	LFT	RSM	WRUP	Heur-Best	CBA-Best	WRUP	LFT	MINSLACK	OC
78 PROBLEMS WITH KNOWN OPTIMAL SOLUTIONS										
AVG ± σ	4.69±	4.64±	6.66±	5.42±	3.32±	1.83±	2.25±	3.72±	3.69±	3.62±
(%)	4.22	4.15	6.29	5.34	3.79	2.83	3.21	3.93	3.85	4.48
OPT (%)	21.79	21.79	16.66	24.36	34.61	56.41	51.28	29.49	34.62	38.46
GLOBAL BEST(%) (1.54 ± 2.70 %)	30.76	30.76	23.07	34.61	55.12	92.31	79.48	44.87	47.44	57.69
CBA-BEST (%)	-	-	-	-	-	-	89.74	48.72	51.28	64.10
92 PROBLEMS (78 OF THEM WITH KNOWN OPTIMAL SOLUTIONS)										
AVG ± σ	4.87±	4.71±	8.87±	6.13±	3.44±	1.82±	2.22±	3.77±	3.89±	4.00±
(%)	4.08	4.04	8.55	5.78	3.70	2.74	3.15	3.79	3.68	4.54
GLOBAL BEST(%) (1.57 ± 2.62 %)	23.91	27.17	18.47	29.35	48.91	92.39	78.26	43.48	42.39	50.00
CBA-BEST (%)	-	-	-	-	-	-	88.00	46.74	45.65	50.00

Another point to be noted is that problem size and resource characteristics do not seem to affect CBA's performance and the distribution of optimal solutions found by the method is uniform and nondiscriminative among tight or loose resource constraints.

The performance of CBA is also consistent in the 14 large size difficult problems with low complexity. When compared to Heuristic-Best results of optimal solution deviation CBA-Best results are at least twice as good. The maximum deviation from best branch and bound solution is 5.88 per cent for CBA-WRUP whereas the same measure is 12.15 per cent for LFT. In these problems the poor performance of WRUP without CBA is due to the fact that it becomes a rule based on only resource requirements when complexity is too low. On the other hand, CBA's consistency in good results lies in its grasping the nature of the problem and thus carrying all heuristic intuition within its methodology. The heuristic rules regard the problem from one viewpoint only, i.e., slack values or latest finish times. When the latter measures are the same for all activities in the resource conflict set, then dispatching rules break ties arbitrarily. An example is given below:

Activity	est	lft	p	slack
i	5	12	5	2
j	5	9	2	2

In this example where i and j are the only activities in the conflict set MINSLACK rule decides arbitrarily whereas CBA schedules j before i according to condition A1. In this case CBA decides in the

same way as LFT and RSM, as well as taking slack time measure into consideration. This is the advantage of CBA logic over heuristic rules. RSM performs rather poorly when compared to CBA although it is based on a similar notion of pairwise conflicts. The performance is poor because concepts of groupwise conflicts and minimal conflict sets do not exist in RSM rule and only pairwise conflicts are dealt with.

There exists a major disadvantage in the application of CBA. In the procedure when no conflicts occur among activities at a certain scheduling decision point (Case C), the rule acts greedily and assigns all of these activities at that time. This type of action leads to the drawback of neglecting possible pairwise conflicts of the present time nonconflicting activities with the remaining unassigned activities. This is one of the causes preventing CBA from obtaining better solutions. A measure against the latter logical flaw might be some kind of look ahead rule checking future pairwise conflicts. However, the localization of essential conditions would then be lost and the time efficiency of the method would be doubtful.

The second point is the employment of heuristic rules when at a certain instance the conditions are not tight enough to identify precedence relationships. This type of situation takes place when either the time constraints are too tight or too loose. With the optimally solved problems a notable improvement is observed when WRUP is employed with CBA and about 90 per cent of the time CBA+WRUP rule obtains the best results among CBA. There is also some improvement when

LFT is used with CBA against LFT used alone. Referring to the observation of CBA+WRUP against WRUP the latter fact demonstrates that CBA does not revert to heuristic rules very frequently or uses these rules after establishing some, if not all, precedence relationships among conflicting activities. CBA requests heuristic aid at about 5 per cent of the time, i.e., the ratio of the number of times a heuristic is employed to the number of iterations is about 0.05.

The third point is the issue of establishing CBA-best results after using CBA with four heuristic rules. A problem must be run thirteen times(ten times for WRUP and three times for LFT,OC,MINSLACK) to obtain CBA best solution.

At this point it is appropriate to discuss CPU times for CBA and heuristic rules. A single CBA run takes on the average 1, 10, 20 CPU sec. for 25, 51 and 103-activity problems respectively on a 70/86 PS2 IBM PC computer. The heuristic rule LFT takes 0.5, 7, 12 CPU sec. on the average for the corresponding problem sizes. WRUP takes on the average half of the CPU times taken by LFT since it is a list priority rule. On the other hand RSM takes four or five times the CPU time of CBA. It should be reminded that the CPU times include I/O times. If a comparison is to be made with the previous research [6] LFT and other heuristic rules are claimed to .take about 3 sec. for 103 activity problems on the BULL DPS8/49 computer.Since a single CBA run takes twice as much time as LFT and has to run thirteen times then CBA approach can be reported to to take 78 CPU sec. on a faster computer, BULL DPS8/49.

These results are true for the above specific 92 problems whereas for other problems of similar sizes, i.e. 70-80 activity, 8-10 resource type problems, CBA seems to be very fast taking about 2-3 CPU sec. on the 70/86 PS2 computer in each run. This contaradictory situation is thought to be due to the existence of dummy activities created by the transformation of the 92 problems into activity-on-arc networks. To summarize, the CPU time of local CBA is reasonable for large size problems and implementation in quasi-dynamic environments is therefore feasible.

VII. TRANSFORMATION OF JOB-SHOP PROBLEMS INTO PROJECT SCHEDULING PROBLEMS

7.1. Specification of Job-Shop Problems

Job-shop scheduling problems constitute a proper subset of resource-constrained project scheduling problems and can be easily transformed to their format. The job-shop problems considered here are of classical type in the sense that a machine is a single-purpose one and can handle one operation at a time. In other words, machine-sharing and routing flexibility are not allowed. Furthermore, process times are sequence independent and deterministic nonpreemptive scheduling is assumed.

However, the former assumption of single-purpose machines handling one operation at a time is not very resrictive and can be dealt with nonintegral machine utilization rates and choice representation in the resource types required by an operation. The decision rules used in project scheduling are based on activities(operations) only, but machine considerations can be easily incorporated into these rules in case alternate routing is allowed. When specifically considering CBA implementation on these problems, it is not difficult to handle routing flexibility in the statements of essential conditions. To achieve this kind of flexibility the definitions of conflicts should be updated and a look-ahead rule should be added with respect to machines to avoid bottlenecks.

There are no restrictions with respect to release times, precedence among jobs, availability of identical machines and also to

the utilization of the same machine more than once by a job(serial or nonserial routing).

7.2. Transformation of Job-Shop Problems to Activity-On-Node Project Networks

An n job, L machine general($a_m \geq 0$, precedence allowed) job-shop problem can be transformed to a project scheduling problem by the following procedure:(The notation is found in the Appendix)

Step 0. Create a dummy starting node and an ending node. The number of resource types is L. ($R_l \geq 1$, l=1..L.)

Step 1. Create a dummy node(operation) for each job with no predecessors and link these nodes to the dummy starting node.($r_{im1}=0$, l=1..L; $p_{im}=a_m$, m$\in${No predecessor set}).

Step 2. Starting from the first operation of each job create serial paths of nodes representing each job's operation sequence with their corresponding resource requirements and process times. Go to Step 3a if precedence exists among jobs. Otherwise go to Step 3b.

Step 3a. Link the final node of each job to the dummy ending node if no successor jobs exist. Go to Step 4.

Otherwise, link each predecessor's final node to its corresponding successor's initial node, representing the precedence relationships by the necessary number of dummy arcs. Go to Step 4.

Step 3b. Create n dummy nodes(operations) for each job. (r_{im1}=0, l=1..L; p_{im}=D_{max} - d_m, m=1..n). Link these dummy nodes to the final ending node. Stop.

Step 4. Repeat Step 2 and Step 3 until all jobs are represented in the network. Stop.

In the above procedure job-shops with no precedence are represented as n serial parallel paths with n dummy activities for nonzero release times and n dummy activities for individual due dates.

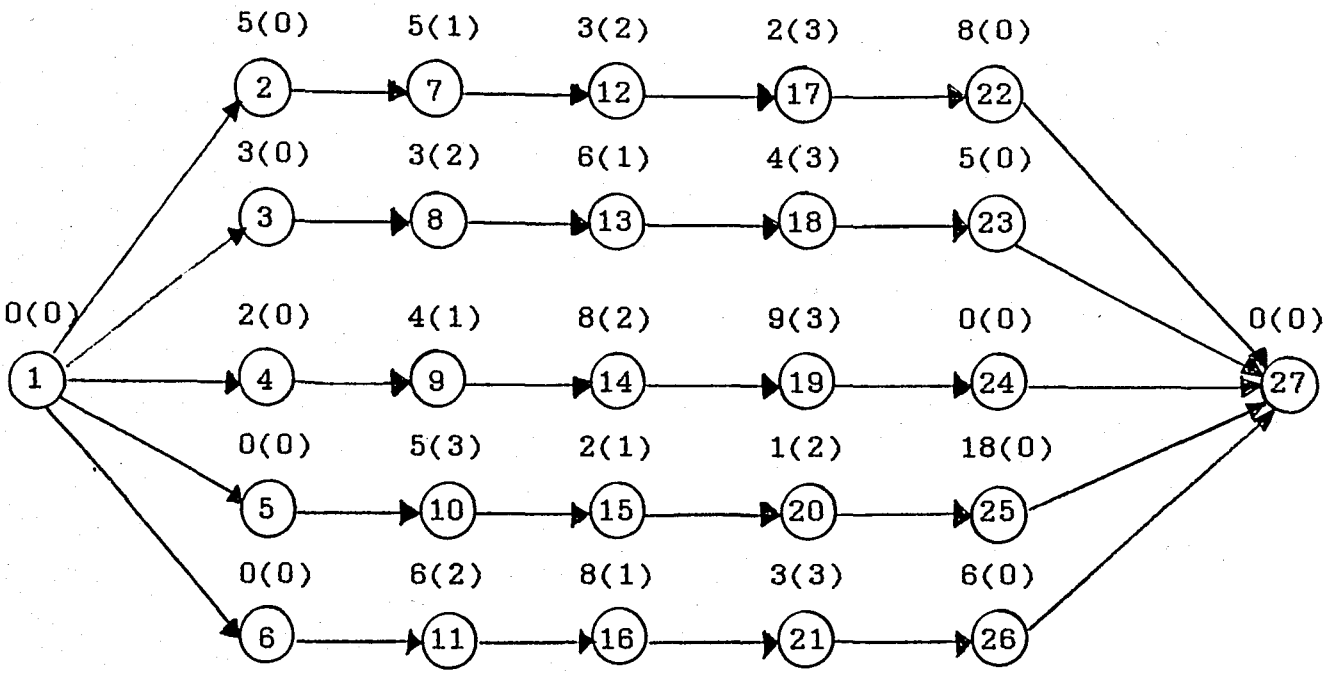
On the other hand, for job-shops with precedence the number of parallel paths are bounded from above by n and the dummy activities required are the ones representing nonzero release times. In the job-shop representation the resource availability of dummy resource type zero which represents the arrival times is assumed to be unlimited.

This is due to the fact that individual due dates are superfluous in the case of precedence among jobs and an overall due date for all jobs is assigned.

Example 7.1. (Without precedence)

J_m	p_{im}			Technological Order	a_m	d_m
	M1	M2	M3			
1	5	3	2	M1/M2/M3	5	22
2	6	3	4	M2/M1/M3	3	25
3	4	8	9	M1/M2/M3	2	30
4	2	1	5	M3/M1/M2	0	12
5	8	6	3	M2/M1/M3	0	24

D_{max} = 30



Notation: duration(type of machine required)

FIGURE 7.1. Activity-on-node diagram for example 7.1.

Example 7.2. (With precedence)

J_m	p_{im}			Technological Order	a_m	Predecessor
	M1	M2	M3			
1	1	3	5	M1/M2/M3	0	-
2	5	8	6	M2/M1/M3	2	-
3	7	0	7	M1/M3	-	2
4	5	4	0	M2/M1	-	1
5	4	3	5	M1/M2/M3	7	-
6	1	6	3	M3/M1/M2	-	4
7	4	3	5	M3/M2/M1	-	4
8	6	0	4	M3/M1	-	3

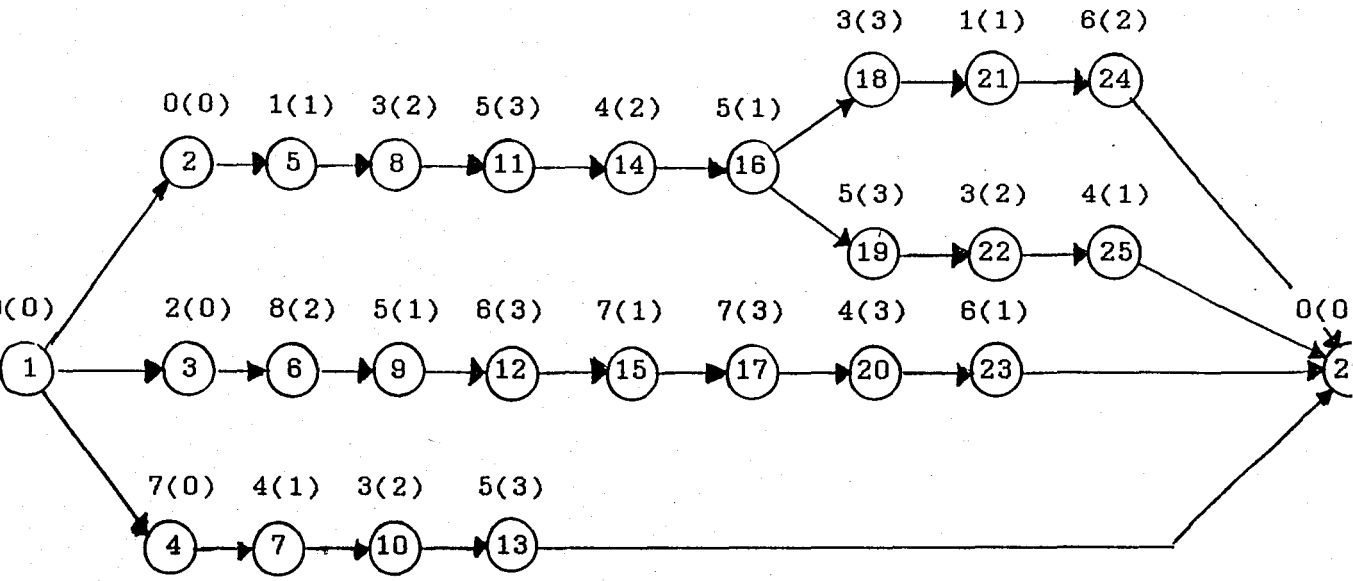


FIGURE 7.2. Activity-on-node diagram for example 7.2.

VIII. PERFORMANCE CRITERION IN JOB-SHOP SCHEDULING

One of the major problems in job-shops concerns due date achievement. Minimization of makespan becomes an objective of lesser importance when compared to the accordance of schedules with due dates. The current statements of the essential conditions in CBA provides the possibility of their utilization and evaluation with respect to maximum tardiness, T_{max}, criterion without making any changes in the condition statements.

No precedence job-shop problems which are transformed to project scheduling problems are solved by CBA in project scheduling terms with respect to T_{max} criterion as follows:

With the due date arrangements made during the transformation each job has a dummy activity with $p_{im}= D_{max}-d_m$ appended to its final operation and the ending node's lft is assigned $\max\{D_{max},lft_{cpm}\}$ which is the lft obtained by forward CPM.

Getting on with backward CPM the final operation of each job has:

$$lft'_m=D_{max}-(D_{max}-d_m)=d_m, \qquad m=1..n.$$

The above method of incorporating due dates into lft_{cpm} and backtracking from D_{max} cause the updated operation lft'_{im} for all jobs to be viewed as operation due dates which are in turn considered in pairwise comparisons and infeasibility checks.

As observed in the following infeasibility check,

$$\alpha' = \max\{\, 0,\ -lft'_{max} + t_{now} + \sum_{k \in SET} p_{ik} \}$$

$$= \max\{\, 0,\ -d_{max} + t_{now} + \sum_{k \in SET} p_{ik} \}$$

the penalty is only valid for a project duration violating d_{max} which is the maximum operation due date over the operation pair. In the case of a violation of operation due dates job due dates are also violated since each job is a series of operations. If $\alpha > 0$, then all due dates are updated by α which is the correct action for T_{max} criterion.

Similarly, in the pairwise comparisons updated lst'_{im} values represent the latest start time with respect to job due dates and the comparisons are evaluated accordingly.

The above adjustments enable CBA to solve job-shop problems without precedence with respect to T_{max}.

Considering the problems with precedence the procedure is similar to original project scheduling problems since individual due dates are assumed not to exist. Consequently, an overall due date, D, is assigned to the final node instead of lft_{cpm} . It should be noted that the forward CPM lft does not incorporate individual due dates in this case. Backward CPM is then applied to the network starting from the value D. Again operation lft'_{im} are to be regarded as operation due dates but in this case all operation lft'_{im} differ by the same amount "$D-lft_{cpm}$" from their corresponding original forward CPM lft_{im} whereas in the former case individual due dates affect and vary the difference between the operations' forward CPM lft_{im} and updated lft'_{im} values.

IX. DUE DATE ASSIGNMENT RULE AND HEURISTICS FOR JOB-SHOP PROBLEMS

9.1. Due Date Assignment

In this study jobs have their due dates based on their Total Work Contents(TWK). This type of assignment is common in job-shop literature [21,24,31] and furthermore, the heuristics to be compared against CBA are reported to perform their best under TWK rules [21,24]. Additionally, the multiplier, k, in TWK rules does not seem to be effective on heuristic performance as long as it is not too large[21,24].

Consequently, the due date expression for each job in no precedence job-shops is as follows:

$$d_m = a_m + k*TWK_m$$
$$= a_m + k*(\sum_{j_{im}} p_{im})$$

On the other hand, for job-shops with precedence the overall due date D is found by a multiple of CPM length:

$$D = k * CPM$$

The choice of the multiplier k in both cases depend on the network/resource characteristics of the problems. From previous research [20] it is known that certain characteristics such as Resource Utilization Factor(UF) and Dominant Obstruction Value(DOV) constitute a measure of project duration elongation over CPM duration. High values of UF and DOV when accompanied by the low complexity characteristic of job-shop problems lead to about 40 - 50 per cent deviation from CPM length.

Using the above information and the fact that the test problems created belong to the medium tightness class with respect to resource characteristics, the multiplier k is assigned a value of 1.2.

9.2. Heuristics To Be Compared with Local CBA in Job-Shop Problems

The heuristics which constitute a comparison background for CBA performance are selected according to their evaluation in previous research. According to this criterion Least Slack(SLACK(JOB), Earliest Operation Due Date(EDD(OPN)), Least Slack per Operation(SLACK/OPN), Truncated Shortest Imminent Operation(SI^x) are the most well known and successful rules as given by a research by Blackstone et al. [24]. Modified Operation Due Date(MOD) is also a promising rule as stated by Baker and Kanet [25] although it has not been tested against a large number of rules. Finally, a rule based on processing times, Longest Remaining Processing Time(LR) is also selected as a worst-performing rule to complete the comparison spread.

The point in the selection of the above rules is that due date based rules such as SLACK(JOB), EDD(OPN), SLACK/OPN and MOD have been shown to be most effective under TWK due date assignment rules for minimizing tardiness criteria. SI^x rule is also claimed to be the one of the most promising rules among SI rooted rules with respect to the same criteria since it incorporates due date considerations. Furthermore, SI rules are claimed to perform well under tight due dates which is the case in this research.

The detailed definitions of the selected rules are given below:

SLACK(JOB): Choose the operation with the lowest value of Z_{im}:

$$Z_{im} = d_m - t_{now} - \sum_{j_{im} \in S_m} p_{im}$$

where S_m: Set of all operations of J_m which are still to be completed.

EDD(OPN): Choose the operation with the lowest value of Z_{im}:

$$Z_{im} = d_{im}$$

where d_{im} is the operation due date.

SLACK/OPN: Choose the operation with the lowest value of Z_{im}:

$$Z_{im} = (d_m - t_{now} - \sum_{j_{im} \in S_m} p_{im}) \; / \; \sum_{j_{im} \in S_m} p_{im}$$

SI^x: Choose the operation with the lowest value of Z_{im}:

$$Z_{im} = \begin{cases} p_{im} & \text{if } (d_m - t_{now} - \sum_{j_{im} \in S_m} p_{im}) \geq 0, \; m=1..n. \\ d_m - t_{now} - \sum_{j_{im} \in S_m} p_{im} & \text{o/w.} \end{cases}$$

(In the original statement of this rule a term accounting for job delay exists. This term is omitted here.)

MOD: Choose the operation with the lowest value of Z_{im}:

$$Z_{im} = \max\{ d_{im}, \; eft_{im} \}$$

LR: Choose the operation with the lowest value of Z_{im}:

$$Z_{im} = - \left(\sum_{j_{im} \in S_m} p_{im} \right)$$

It should be pointed out that the heuristics above solve the job-shop problem in the classical job-shop context. Job-shop heuristics such as SLACK/OPN and LR use job identification which is valuable

information. EDD(OPN) and MOD do not require job identification since they use operation due dates.On the other hand, CBA solves the problems using operation information only.

X. EXPERIMENTAL DESIGN FOR JOB-SHOP PROBLEMS

The test problems include six types of problems the list of which are found in Table 10.1., varying from classical job-shop problems to quite general nonrestrictive types. The classification of test problems into types aims at specifying changes in the performance of CBA corresponding to different types.

The problems are grouped into two main classes: Problems with Precedence relationships among jobs and problems without. The two main groups in turn are classified into three subcategories: (i) Serially-routed - resource limit=1; (ii) nonserially-routed - resource limit=1; and (iii) nonserially-routed - resource limit≥1 categories. These categories range from restrictive to less restrictive. Serial routing implies that each job is processed on each type of machine once. When machine availability exceeds unity, identical machine groups are assumed and the machines are selected arbitrarily if more than one machine of the same type are free to process an operation.

Each type of problem consists of 15 test problems totalling to 90 problems in all. To keep the problems as uniform as possible and to determine the proper multiplier k, valid for all problems, network/resource characteristics of all problems are attempted to be kept at the same levels. The UF values range between [0.5-1.1] and DOV values range between [0.0-0.6]. Complexity ratio lies between [1.1-1.2]. The maximum number of machine types is 9 and the maximum number of operations is 84. On the average a problem consists of 60

operations and 5 machine types and the average number of operations/job is 10 in no precedence problems and the same number is 3 in problems with precedence. Release times of jobs lie between [0-10] and process times between [1-15]. In all types of problems machine utilization is as uniform as possible among different machine types.

TABLE 10.1. Characteristics of problem types.

Type	Definition		
	Precedence	Serial routing	resource limit=1
I	-	x	x
II	-	-	x
III	-	-	-
IV	x	x	x
V	x	-	x
VI	x	-	-

"x" = "characteristic exists"

XI. DISCUSSION OF JOB-SHOP RESULTS

The results obtained by CBA and their comparison to six heuristic rules indicate that CBA is a favorable method in job-shop context also. Table 11.1. consists of the average T_{max} values and their corresponding standard deviations for each problem type and also overall results. Best solution finding percentages and the percentage of obtaining CBA-Best solutions for CBA-OC, -LFT, -WRUP, -MINSLACK are also stated in Table 11.1.

The results of Duncan multiple range tests on type(treatment) pairs for each heuristic rule and CBA options are found in Table 11.2.

The closest follower of CBA with respect to T_{max} performance is EDD(OPN) rule. This is also true for project scheduling problems where LFT rule which is equivalent to EDD(OPN) is the most challenging rule against CBA. For the total of 90 problems EDD(OPN)'s average T_{max} value is 15.51 per cent worse than CBA-OC's value which represents the best of the four CBA options. It is observed that for job-shop problems CBA-OC performance is slightly better than CBA-WRUP performance, which is not the case for project scheduling problems. This fact is also evident when considering CBA-Best finding percentages. For job-shop problems CBA-WRUP loses the advantages it has in project management since the complexity of job-shop problems is very low and the insight WRUP gains through immediate precedence relationships is thus lost. Yet, CBA-WRUP closely follows CBA-OC in all types of problems. On the other hand, the set logic found in CBA-OC is very effective in this context of parallel

path networks.

Comparing the rest of the dispatching rules with CBA-OC, Table 11.1. demonstrates that CBA-OC outperforms them in the range [15.51 - 86.08] per cent. The worst performance belongs to LR rule as expected. Although MOD is the next successful among dispatching rules after EDD(OPN) it is a distant follower of CBA-OC since its performance is 29.34 per cent worse. Best finding percentages are also observed to be in favor of CBA.

While scrutinizing Table 11.1.in more detail, CBA, although never out of the best position among all for each type of problem, seems to be comparatively more successful when applied to job-shops with precedence. The range of outperformance is [9.30 - 75.31] per cent in the problems without precedence whereas the same range is between [33.50 -177.19] per cent in problems with precedence. Thus, there exists a significant difference in the range of outperformance. At this point it is suitable to evaluate the information in Table 11.2. The data in Table 11.1. have undergone Duncan multiple range tests with the aim of specifying significant differences between type means. Naturally, prior to the application of Duncan tests one way ANOVA tables have been prepared for all heuristics and CBA options to discover that type effects are highly significant at 5 per cent and 1 per cent significance levels.

It is observed in Table 11.2. that for all heuristics and CBA options the common characteristic is that no precedence type means are

significantly different from precedence type means. Precedence type means are not different from each other and the same is true for no precedence type means. This observation implies that routing and the existence of machine groups are not as effective on T_{max} criterion as precedence among jobs.

The consistency of the above observation on both tables implies that although CBA outperforms all heuristics in all cases its success is more pronounced in job-shops with precedence. It should be reminded that the precedence structure in the test problems is not restricted in any way.

With respect to time efficiency of CBA in solving job-shop problems, the maximum CPU time observed is 3 CPU sec. on 70/86 IBM PS2 for the largest sized problem of 84 operations and 9 resource types. CBA seems to take almost the same CPU time as the heuristic rules in solving the job-shop problems while outperforming them all. Additionally, since CBA-OC option is the best performing option in the job-shop context, a single run is sufficient to obtain a good CBA result.

The pleasing performance of CBA through its conflict based logic is to be appreciated in job-shop context since CBA accomplishes this achievement by using operation information only and solving problems in project scheduling terms. As a local condition implementer, CBA disregards job identification, yet incorporating job specifications such

as job remaining work content, due date etc. through operation lft and lst values.

In conclusion, CBA performance is found to be considerably successful in job-shop context, both for job-shops without precedence and with precedence. This is an expected result since it is noted that CBA's successful performance is also consistent in low complexity large size project scheduling problems where heuristic rules become erratic in their performance. Furthermore, the local essential conditions which constitute CBA logic may incorporate problem specific characteristics, specifically in the context of resource utilization.

TABLE 11.1. Average T_{max} results obtained by heuristics and local CBA.

RULE:	EDD (OPN)	SLACK/ (OPN)	SLACK (JOB)	SI*	LR	MOD	HEUR-BEST	CBA-BEST	CBA WRUP	CBA LFT	CBA MINSLACK	CBA OC
TYPE I												
AVG±σ	20.47± 11.46	20.53± 13.36	22.73± 12.10	22.67± 10.56	29.06± 11.82	22.67± 11.17	16.86± 9.41	15.81± 10.04	19.93± 12.17	20.46± 11.45	19.60± 10.03	16.85± 9.86
TYPE II												
AVG±σ	18.93± 11.18	27.00± 13.90	26.00± 17.77	25.33± 17.94	29.80± 18.58	20.80± 12.08	17.73± 11.11	16.53± 11.60	18.06± 11.04	20.73± 14.74	21.20± 14.82	18.40± 13.17
TYPE III												
AVG±σ	15.60± 8.28	18.20± 11.91	19.33± 11.26	19.53± 11.79	29.33± 12.16	19.07± 11.40	14.47± 8.66	13.92± 8.81	15.00± 8.89	15.39± 8.12	15.46± 8.48	15.06± 9.17
TYPE IV												
AVG±σ	8.46± 7.64	12.60± 9.21	11.40± 9.10	11.73± 6.29	14.73± 10.77	9.26± 9.30	7.60± 6.44	5.06± 5.89	6.20± 5.84	7.33± 7.06	7.93± 7.59	6.20± 7.16
TYPE V												
AVG±σ	10.47± 6.84	12.33± 8.68	11.26± 8.00	10.20± 7.09	16.13± 10.46	11.33± 8.28	8.53± 7.72	6.40± 7.15	7.66± 7.39	7.26± 7.52	9.14± 7.41	7.66± 8.06
TYPE VI												
AVG±σ	4.26± 5.20	5.13± 6.25	4.80± 5.44	4.53± 5.64	6.80± 6.51	4.40± 6.02	3.80± 5.28	3.00± 4.17	3.53± 4.72	4.20± 5.23	3.66± 4.69	3.53± 4.93
PROBLEMS WITHOUT PRECEDENCE (45 PROBLEMS)												
AVG±σ	18.33± 10.61	22.29± 13.32	22.69± 14.28	22.51± 14.02	29.40± 14.52	20.84± 11.65	16.35± 9.94	15.41± 10.29	17.66± 10.98	18.86± 12.03	18.75± 11.81	16.77± 11.15
% over CBA-OC	9.30	32.91	35.30	34.22	75.31	24.26	6.10	**	5.30	12.46	11.81	*
GLOBAL BEST(%)	42.22	20.00	13.33	4.44	6.66	17.77	62.22	80.00	53.33	42.22	35.55	51.11
CBA-BEST(%)	-	-	-	-	-	-	-	-	64.44	48.88	46.66	64.44

TABLE 11.1. Average T_{max} results obtained by heuristics and local CBA (Continued).

RULE:	EDD (OPN)	SLACK/ (OPN)	SLACK (JOB)	SI^x	LR	MOD	HEUR-BEST	CBA-BEST	CBA WRUP	CBA LFT	CBA MINSLACK	CBA OC
PROBLEMS WITH PRECEDENCE (45 PROBLEMS)												
AVG±σ	7.73± 7.40	10.02± 8.84	9.15± 7.84	8.82± 7.08	12.59± 9.91	8.33± 8.82	6.64± 6.85	4.82± 6.02	5.79± 6.34	6.26± 6.83	6.91± 7.14	5.79± 6.06
% over CBA-OC	33.50	73.05	58.03	52.33	117.13	43.86	37.76	‡‡	00.00	8.11	19.34	‡
GLOBAL BEST(%)	31.11	17.77	22.22	31.11	13.33	31.11	53.33	91.1	66.66	48.88	46.66	60.00
CBA-BEST(%)	-	-	-	-	-	-	-	-	73.33	48.88	48.88	66.66
TOTAL 90 PROBLEMS												
AVG±σ	13.03± 10.76	16.16± 12.94	15.92± 13.42	15.66± 13.14	20.99± 14.91	14.59± 12.02	11.50± 9.89	10.12± 9.86	11.72± 10.74	12.56± 11.64	12.83± 11.39	11.28± 11.02
% over CBA-OC	15.51	43.26	41.13	38.82	86.08	29.34	13.63	‡‡	3.90	11.34	13.74	‡
GLOBAL BEST(%)	36.66	18.88	17.77	17.77	10.00	24.44	57.77	85.55	60.00	45.55	41.11	55.55
CBA-BEST(%)	-	-	-	-	-	-	-	-	68.88	50.00	48.88	65.55

TABLE 11.2. Results of Duncan multiple range tests on pairs of type means.

RULE:	EDD (OPN)	SLACK/ OPN	SLACK (JOB)	SI*	LR	MOD	HEUR-BEST	CBA-BEST	CBA WRUP	CBA LFT	CBA MINSLACK	CBA OC
TYPE PAIR												
I - VI												
I - V												
I - IV		‡										
I - III	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡
I - II	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡
II - VI												
II - V												
II - IV												
II - III	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡
III - VI												
III - V	‡	‡							‡	‡	‡	
III - IV	‡	‡	‡	‡			‡			‡	‡	
IV - VI	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡
IV - V	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡
V - VI	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡

Level of significance = 5%

"‡" = "No significant difference exists between type means"

XII. CONCLUSION

In this study, the resource-constrained scheduling problem is treated with a method which stands midway between optimization and heuristic techniques. Local CBA which is the approach proposed here, considers the objective function and all kinds of constraints imposed on the problem simultaneously. The advantages of the method are the precision of its decisions and the efficiency gained through its localizing aspect.The method is reliable with respect to its solutions in spite of its utilization of dispatching rules when required since heuristic rule employment is quite restricted.

In conclusion, local CBA seems to perform well with regards to quality of its solutions and time efficiency, fulfilling the expectation that it should be used both in static and quasi-dynamic situations. Additionally, the development of local essential conditions contributes to the general resource constrained scheduling problem from the theoretical point of view and these concepts might be extended for other objectives and problem specific constraints. It is believed that local CBA which is implemented on the most general problem classes in this study is a promising area of research leading to its future application on recently defined scheduling problems which incorporate nonrenewable resource constraints and cost based objective functions[33,34,35].

APPENDIX

(NOTATION)

APPENDIX

t_{now} : Current point of time.

$\ll$: "precedes"

AP : Set of activities in progress.

p : Activity duration.

U : Set of unassigned activities.

UA : Set of activities assigned at t_{now}.

FIN : Set of activities finished at t_{now}.

Γ^* : Set of activities which are schedulable at t_{now}.

i : Activity(operation) index.

h : Resource(machine) index.

m : Job index.

J_m : Job m.

j_{im} : Operation i of job m.

$r_{i,h}$: Resource requirement of resource type h required by activity i per unit time.

R'_h : Available resource level per unit time of type h at t_{now}.

R_h : Resource limit for resource type h per unit time.

est_i : Earliest start time of activity i.

lft_i : Latest finish time of activity i.

lft_{max} : $\max_{k=i,j} \{lft_k\}$

lft_{cpm} : Final node's lft obtained by forward CPM.

d_m : Due date of J_m.

a_m : Arrival time of J_m.

D_{max} : Maximum due date over all jobs J_m.

C_i : Completion time of activity i.

SET : Largest set of conflicting activities at t_{now}. (SET⊂r)

SET': Set of conflicting activities remaining after pairwise precedence rule application.

SET" : Set of conflicting activities remaining after groupwise precedence rule application.

|SET| : Number of activities in SET.

S' : Maximal nonconflict set; a set such that if any activity i∈SET" is included in S', S'+i cannot be processed concurrently. (S'⊂SET")

$R^{est}_{h,t}$: Units of resource type h required, based on EST scheduling in time period t.

$pro.dur_{s'}$: Project duration obtained by CPM if S' is scheduled at t_{now}.

$pro.dur_{previous}$: Project duration accepted at the last infeasibility check time.

$ERR_{s'}$: Excess resource ratio if set S' is assigned at t_{now}.

$$ERR_{s'} = \sum_{h=1}^{H} \sum_{t=t_{now}}^{Pro.dur_{s'}} \left[\frac{\max \{ 0, R^{est}_{h,t} - R_h \}}{R_h} \right]$$

$OC_{s'}$: Opportunity cost of assigning set S' at t_{now}.

$$OC_{s'} = ERR_{s'} * \max \{ 0, pro.dur_{s'} - pro.dur_{previous} \}$$

If any one of the multipliers is zero, then the formula becomes:

$OC_{s\cdot} = ERR_{s\cdot} + \max \{ 0, pro.dur_{s\cdot} - pro.dur_{previous} \}$

w_p : precedence weight $(0 \leq w_p \leq 1)$

np_i : number of immediate successors of activity i.

UF : Resource utilization factor.

$$= \max_h \left\{ \frac{\sum_i p_i * r_{ih}}{CPM * R_h} \right\}$$

DOV : Dominant obstruction value.

$= 1.5\ MO$ if $0.5\ MO > MO2$

$= MO + MO2$ if $0.5\ MO \leq MO2$

where $MO = \max_h \{o_h\}$ and MO2 is the second largest o_h.

$$o_h = \frac{\sum_{t=1}^{CPM} \max \{0, R^{**}{}_{ht} - R_h\}}{\sum_i p_i * r_{ih}}$$

BIBLIOGRAPHY

1. Erschler, J., Roubellat, F. and Vernes, J.P. , "Finding some essential characteristics of the feasible solutions for a scheduling problem", Operations Research, Vol. 24, No. 4, pp. 774-783, 1976.

2. Erschler, J., Roubellat, F., "An approach to workshop real time scheduling problems", Advanced Information Techniques for Industrial Material Flow Systems, NATO ASI Series, F53, 1988.

3. Erschler, J. and Esquirol, P., "Decision aid in job shop scheduling:a knowledge based approach", IEEE International Conference on Robotics and Automation, San Francisco, U.S.A., 1986.

4. Bensana, E., Bel, G. and Dubois, D., "OPAL: A multi-knowledge-based system for industrial job shop scheduling", International Journal of Production Research, Vol. 26, No. 5, pp. 795-819, 1988.

5. Christofides, N., Alvarez-Valdes, R. and Tamarit, J.M., "Project scheduling with resource constraints: A branch and bound approach", European Journal of Operational Research, Vol. 29, pp. 262-273, 1987.

6. Alvarez-Valdes, R. and Tamarit, J.M., "Heuristic algorithms for resource constrained project scheduling: A review and an empirical analysis", Advances in Project Scheduling, ed. R. Slowinski and J. Weglarz, ELSEVIER, 1989.

7. Stinson, J.P., Davis, E.W. and Khumawala, B.W. , "Multiple-resource constrained scheduling using branch and bound", AIIE Transactions, Vol. 10, pp. 252-259, 1978.

8. Talbot, F.B. and Patterson, H. , "An efficient integer programming algorithm with network cuts for solving resource constrained scheduling problems", Management Science, Vol. 24, pp. 1163-1174, 1978.

9. Hastings, N.A.J., "On resource allocation in project networks", Operational Research Quarterly, Vol. 23, pp. 217-221, 1972.

10. Willis, R.J. and Hastings, N.A.J., "Project scheduling with resource constraints using branch and bound methods", Operational Research Quarterly, Vol. 27, pp. 341-349, 1976.

11. Pritsker, A.A., Watters, L.J. and Wolfe, P.M. , "Multi-project scheduling with limited resources: a zero-one programming approach", Management Science, Vol. 16, pp. 93-108, 1969.

12. Patterson, J.H. and Huber, W.D., "A horizon varying zero-one approach to project scheduling", Management Science, Vol. 20, pp. 990-998, 1974.

13. Wiest, J.D. , "A heuristic model for scheduling large projects with limited resources", Management Science, Vol. 13, pp. B359-B377, 1967.

14. Thesen, A., "Heuristic Scheduling of activities under resource and precedence restrictions", Management Science, Vol. 23, pp. 412-422, 1976.

15. Elsayed, E.A. and Nasr, N.Z. , "Heuristics for resource constrained scheduling", International of Journal Production Research, Vol. 24, No. 2, pp. 299-310, 1986.

16. Mohanty, R.P. and Siddiq, M.K., "Multiple-projects multiple-resources constrained scheduling: some studies", International Journal Production Research, Vol. 27, No. 2, pp. 261-280, 1989.

17. Davis, E.W. and Patterson, J.H., "A comparison of heuristics and optimum solutions in resource constrained project scheduling", Management Science, Vol. 21, pp. 944-955, 1975.

18. Herroelen, W.S. , "Resource constrained project scheduling-the state of the art", Operational Research Quarterly, Vol. 23, pp. 261-275, 1972.

19. Willis, R.J. , "Critical path analysis and resource constrained project scheduling- theory and practice", European Journal of Operational Research, Vol. 21, pp. 149-155, 1985.

20. Ulusoy, G. and Ozdamar, L. , "Heuristic performance and network/resource characteristics in resource-constrained project scheduling", Operational Research Quarterly, Vol. 40, No. 12, pp. 1145-1152, 1989.

21. Weeks, J.K., "A simulation study of predictable duedates", Management Science, Vol. 25, No. 4, pp. 363-373, 1979.

22. Baker, K.R. and Kanet, J.J., "Improved decision rules in a combined system for minimizing job tardiness", International Journal Production Research, Vol. 22, No. 6, pp. 917-921, 1984.

23. Mizayaki, S., "Combined scheduling system for scheduling for reducing job tardiness in a job shop", International Journal Production Research, Vol. 19, pp. 202-211, 1981.

24. Blackstone, J.H., Phillips, D.T. and Hogg, G.L., "A state-of-the-art survey of dispatching rules for manufacturing job shop operations", International Journal Production Research, Vol. 20, No. 1, pp. 27-45, 1982.

25. Baker, K.R. and Kanet, J.J., "Improved decision rules in a combined system for minimizing job tardiness", International Journal Production Research, Vol. 22, No. 6, pp. 917-921, 1984.

26. Muhlemann, A.P., Lockett, A.G. and Farn, C.K. , "Job shop scheduling heuristics and frequency of scheduling", International Journal Production Research Vol. 20, No. 2, pp. 227-241, 1982.

27. Raman, N., Talbot, F.B. and Rachamadugu, R.V., "Simultaneous scheduling of machines and material handling devices in automated manufacturing", Proceedings of 2nd ORSA/TIMS Conference on FMS: Operations Research Models and Applications, Elsevier Science Publishers, The Netherlands, 1986.

28. Yamamoto, M. and Nof, S.Y., "Scheduling/rescheduling in the manufacturing operating system environment", International Journal Production Research Vol. 23, No. 4, 705-722, 1985.

29. Norbis, M.I. and Smith, J.M., "Two level heuristic for the resource constrained scheduling problem", International Journal Production Research, Vol. 24, No. 5, pp. 1203-1219, 1986.

30. Norbis, M.I. and Smith, J.M., "A multi-objective multi-level heuristic for dynamic resource-constrained scheduling problems", European Journal Operational Research, Vol. 33, pp. 30-41, 1988.

31. Slomp, J.S., Gaalman G.J.C. and Nawijn W.M. , "Quasi-on-line scheduling procedures for flexible manufacturing systems", International Journal Production Research, Vol. 26, No. 4, pp. 585-598, 1988.

32. Dumond, J. and Mabert, V.A., "Evaluating project scheduling and due date assignment procedures: An experimental analysis", Management Science, Vol. 34, No. 1, pp. 101-108, 1988.

33. Slowinski, R., " Multiobjective network scheduling with efficient use of renewable and nonrenewable resources", European Journal of Operational Research, 7, 265-273, 1981.

34. Talbot, F. B.," Resource constrained project scheduling with time-resource tradeoffs: The nonpreemptive case", Management Science, 28(10), 1197-1210, 1982.

35. Patterson, J. H., Slowinski, R., Talbot, F. B., Weglarz, J.," An algorithm for a general class of precedence and resource constrained scheduling problems", Advances in Project Scheduling(Chap.1), ed. R. Slowinski and J. Weglarz, ELSEVIER, 1989.