BRANCH AND BOUND ALGORITHMS FOR

SINGLE MACHINE SCHEDULING WITH

BATCH SET-UP TIMES TO MINIMIZE TOTAL

WEIGHTED COMPLETION TIME

by

H.A.J. CRAUWELS\*

A.M.A HARIRI\*\*

C.N. POTTSt

and

L.N. VAN WASSENHOVEtt

95/70/TM

t PPrrooffeessssoorr aatt KINinSgE AAbDd,u Bl-oAuzleizv aUrndi vdeer Csiotyn,s Jtaendcdea,h F 2o1n4t1a3in, eSbalueadui A77ra3b0i5a .Cedex, France.

Professor at KIHDN, J. De Nayerlaan 5, 2860 Sint-Katelijine-Waver, Belgium.

\*\*

Professor at University of Southampton, Southampton S017 1BJ, UK.

t

A working paper in the INSEAD Working Paper Series is intended as a means whereby a faculty researcher's

thoughts and findings may be communicated to interested readers. The paper should be considered

preliminary in nature and may require revision.

Printed at INSEAD, Fontainebleau, FranceBranch and Bound Algorithms for Single Machine

Scheduling with Batch Set-Up Times to Minimize

Total Weighted Completion Time

H. A. J. Crauwels

KIHDN

J. De Nayerlaan 5

2860 Sint-Katelijne-Waver, Belgium

A. M. A. Hariri

Department of Statistics

King Abdul-Aziz University

Jeddah 21413, Saudi Arabia

C. N. Potts

Faculty of Mathematical Studies

University of Southampton

Southampton 5017 IBJ, United Kingdom

L. N. Van Wassenhove

INSEAD

Boulevard de Constance

77305 Fontainebleau, France

July, 1995

Abstract

This paper presents several branch and bound algorithms for a single machine scheduling

problem with batching. Jobs are partitioned into families, and a set-up time is necessary when

there is a switch from processing jobs of one family to jobs of another family. The objective is to

minimize the total weighted completion time. A lower bound based on Lagrangean relaxation

of the machine capacity constraint is derived. Also, a multiplier adjustment method to find

values of the multipliers is proposed. Computational experience with instances having up to 70

jobs shows that the lower bounds are effective in restricting the search.

1 Introduction

Many practical scheduling problems involve processing several families of related jobs on common

facilities, where a set-up time is incurred whenever there is a switch from processing a job in

one family to a job in another family. To avoid set-ups, several jobs of the same family may be

scheduled contiguously to form a batch. A schedule defines how batches are formed, and specifies

the processing order of the batches.

1As an example of a problem that involves scheduling and batching, Ahn and Hyun [1] describe

an application in the manufacture of steel pipes. The manufacturing process for pipes requires the

use of a rolling machine, where the roller to be used depends on the outer diameter of the pipe. A

change in the roller corresponds to a set-up. Thus, orders for pipes with the same outer diameter

lie within the same job family in our model. A review of algorithms and complexity for various

scheduling problems that involve batching is given by Potts and Van Wassenhove [14].

In this paper, we consider a single machine scheduling problem with N jobs which are parti-

tioned into F families. Set-up times are sequence independent: each depends only on the family

of the job to be processed next. We assume that all the jobs are available at time zero and that

each job has a given processing time and an associated positive weight. We wish to find a schedule

which minimizes the total weighted completion time of the jobs.

When all set-up times are zero, the problem is solved in 0(N log N) time by sequencing the

jobs in shortest weighted processing time (SWPT) order (non-decreasing order of processing time

to weight ratios). Monma and Potts [11] show that, for non-zero set-up times, jobs within each

family are sequenced in SWPT order in an optimal schedule. Using this property, they derive

a dynamic programming algorithm which requires O(F2N2F) time. More efficient dynamic pro-

gramming algorithms have been developed subsequently. For the case of unit weights, Ahn and

Hyun [1] develop an algorithm which requires 0(F2NF) time, and Ghosh [5] proposes an O(F2NF)

algorithm for arbitrary weights. Thus, for fixed F, the problem is polynomially solvable, although

the algorithms provide a practical method of solution only when F is very small. The issue of

whether the problem is NP-hard for arbitrary F is unresolved, even for the case of unit weights.

Heuristic methods are also the subject of some studies. For the case of unit weights, Gupta [6]

devizes a SPT-based heuristic. Essentially, it is a single-pass greedy procedure that requires

0(N log N) time: the job to be processed next is chosen so that it has the smallest comple-

tion time amongst all candidate jobs. Although having the advantage of modest computational

requirements, the quality of the schedules that it generates is often rather poor.

Various local search heuristics are also proposed in the literature. Ahn and Hyun [1] develop a

descent (or iterative improvement) heuristic. They use a neighbourhood in which blocks of jobs of

the same batch are shifted from one part of the sequence to another, but with the restriction that

the SWPT ordering of jobs within a family is maintained. Mason [9] proposes a genetic algorithm

which uses the observation that knowledge of which job starts each batch enables a solution to

be constructed by ordering the batches using a generalization of the SWPT rule. Thus, solutions

can be represented as binary strings to which standard genetic operators can be applied. Crauwels

et al. [3] develop simulated annealing, threshold accepting and tabu search methods which build

on the work of Ahn and Hyun. In computational tests, Crauwels et al. find their tabu search

algorithm to be superior to a multi-start version of the descent method of Ahn and Hyun, to

simulated annealing and threshold accepting, and also to Mason's genetic algorithm.

Mason and Anderson [10] propose a branch and bound algorithm. A special feature of their

algorithm is the extensive use of dominance rules to restrict the size of the branch and bound search

tree. They use a forward branching rule which yields partial schedules in which jobs are sequenced

in the initial positions. Their lower bound is derived using objective splitting: the total weighted

2completion time can be partitioned into contributions from the processing times and from the

set-up times, which are optimized separately. The SWPT rule minimizes the first contribution. A

generalized SWPT rule applied to families, where the processing time of a family is replaced by its

set-up time, minimizes the second. Computational results indicate that this algorithm represents

a practical method of solution provided that there are no more than about 30 jobs. The restriction

to fairly small instances is attributed to the weakness of the lower bound: the dominance rules are

the major contributors to the pruning of the branch and bound search tree.

In spite of significant recent research activity in the area of scheduling models which involve

batching, there have been few attempts to design computationally effective enumerative algorithms.

This is unfortunate in view of the practical importance of these problems. In this paper, we propose

a new branch and bound algorithm which is superior to that of Mason and Anderson [10]. It uses

lower bounds that are derived from a Lagrangean relaxation of machine capacity constraints. A

special feature is the use of a multiplier adjustment method for obtaining values of the multipliers.

The ability to construct multipliers with some desirable properties reinforces the view that it is often

possible to exploit problem structure in a Lagrangean procedure, rather than resort to the heavy-

handed subgradient optimization approach [2,7,12,13]. Our computational results demonstrate

that the quality of the lower bounds obtained from the multiplier adjustment procedure is high.

In Section 2, we give a formal statement of our problem and present some dominance crite-

ria. Section 3 derives our lower bound using a Lagrangean relaxation of the machine capacity

constraints. Our multiplier adjustment procedure for finding values of the multipliers is also pre-

sented. Section 4 gives a heuristic method for scheduling the jobs. Branch and bound algorithms

are described in Section 5. Section 6 reports on computational experience with the branch and

bound algorithms, and some concluding remarks are given in Section 7.

2 Problem structure and dominance rules

To state our problem of scheduling with batch set-up times more precisely, we are given a set .A1

containing N jobs that are divided into F families. Each family f, for f = 1, , F. contains n

jobs. All jobs become available for processing at time zero and are to be scheduled on a single

mmamcheidniea.t eFlyo ra tfhteer ia't hjo jbo b in o af d<faipfmf,eifrl,eyfn 1ftw ,f a,wfm,hfii lcfyho. rwA fel s= od e,1 na,on t ein bitFyia .(li ,s eft)-,u lpe tt ipmi fe ds fe niso tree qitusi rperdo cife sas jionbg ftrimome

and wif its positive weight. We assume that the jobs within each family are indexed in SWPT

order. Thus, pi f /wif <

A sequence independent set-up time s f > 0 is incurred whenever a job in family f is processed

i

family f is processed first on the machine. Thus, the processing of each batch must be preceded

by the relevant set-up.

We first present two fundamental results on the structure of an optimal schedule.

Theorem 1 (Monma and Potts [11]). In any optimal schedule, jobs within each family are se-

quenced in SWPT order.

From Theorem 1, the problem reduces to one of merging SWPT-ordered lists of jobs for the

different families to form a sequence.

3 Consider any schedule S = (B1 , . , B„), where B, (v = v) is a batch comprising a max-

imal consecutive subsequence of jobs from the same family f„. We define the weighted processing

timhee f (oWlloPwTi)n gra rteiosu flot rg eivaecsh ab gaetcnhe rBal,i zteod b SeWPTi ,rfu)EleB vf or batch(ie.fs) EthB a.t uses WPT ratios. (1)

WPT(Bo ) = (8 f, E p, f) I E

(

T

Theorem 2 (Mason and Anderson [10]). In any optimal schedule, batches are sequenced in non-

decreasing order of WPT ratios.

We now describe some dominance rules that allow us to restrict the set of candidates for

sequencing next in the continuation of some initial partial schedule S = (B1 , ... , B,), where B,

= 1,... ,v) is a batch. These rules are based on the results of Mason and Anderson [10]. We

assume that S satisfies the conditions of Theorem 1. Let mf denote the number of jobs of family

f (f = 1,...,F) that are contained in S, where 0 < mf <

Suppose that B, is a batch of some family g, where g = f„, and that job (mh -(- 1,h), where

h g, is a candidate to be sequenced immediately after the last job of S. If mh 0, let job (mh , h)

be contained in batch Bu , where 1 < u < v and h = f,.

Our dominance rules are as follows.

Rule 1. Suppose that mg < ng and WPT(Bo ) > pmg+i ,g/wrng+i ,g. Then job (m9 + 1,g) must be

scheduled immediately after the last job of S.

Rule 2. Suppose that v > 1, and WPT(Bv\_i ) > WPT(Bv). If mg = n9 , then S cannot form the

start of an optimal schedule. If mg < n9 , then job (mg + 1,g) must be scheduled immediately

after the last job of S.

f Ei=fm, +1 pi f)I Ehi=im, +1 wif}. If mg =

Rule 3. Suppose that WPT(B,) > minfE{plmf, <, j , } {(s

ng , then S cannot form the start of an optimal schedule. If mg < ng , then job (mg + 1,g)

must be scheduled immediately after the last job of S.

Rule 4. Suppose that mg < ng and p„„±1,h1w,h,4. 1,h > pmg+1,9/wm9+1,g . Then there is no optimal

schedule in which job (mh + 1,h) is scheduled immediately after the last job of S.

Bu) > Prah+1,h1 Wmh+1,h, where v=u+i i,f E)EB . Pif))/ v.=. u.• +.• .1• BBO)E o rw WiPfI.ABo-f-t,

Rule 5. Suppose that mh > 0 and either p,,,,h/wm,,h >

,

hen there is no optimal schedule i n whEich (jSolby (mh + 1, h) is scheduled immediately after

WPTABv+1,...,Bv) = (sh

(

T

the last job of S.

4We now justify the validity of these rules. Rules 2 and 3 follow from Theorem 2. Under the

conditions of Rule 2, schedule S is not consistent with this WPT ordering unless further jobs are

added to batch B. Rule 3 considers an upper bound on the smallest WPT value that can occur

when batches of the unscheduled jobs are formed. If WPT(B.„) exceeds this upper bound, further

jobs must be added to batch B„ in an attempt to satisfy the WPT ordering. The validity of Rules

1, 4 and 5 follow from Corollary 4.1, Corollary 4.2 and Theorem 4 of Mason and Anderson [10],

respectively.

Rules 1. 2. 4 and 5 are each used by Mason and Anderson [10] in their branch and bound

algorithm. However, Rule 3 is not used in any previous studies.

A dynamic programming dominance rule is also useful. For two initial partial schedules Si and

S2 which have identical final jobs and which contain the same jobs, if 52 has a completion time

of its last job and a total weighted completion time that are no smaller than the corresponding

values for Si , then S2 is dominated by S1.

Unfortunately, Rules 1 through 5 and the dynamic programming dominance rule can be used

within the branch and bound search tree only when a forward branching rule is used. Attempts

to make important decisions early through the use of a flexible branching rule have to be balanced

against the loss of these dominance rules.

A useful preprocessing routine is suggested by Rule 1. Suppose that of > 2 and (s f d-pif)/wif >

P2f /W2 f for some family f. Rule 1 ensures that no optimal schedule contains a batch comprising

only job (1, f). Thus, jobs (1,f) and (2,f) must be scheduled contiguously. Using the theory of

f p2 f and

Lawler [8], they can be combined to form a single composite job with processing time pi

weight vii f w21 . Although this composition increases the total weighted completion time by a

constant term p21w11 , an equivalent problem results. After reindexing the jobs, this preprocessing

step is reapplied. Mason and Anderson [10] show that if =- p,fitllif , where (i, f) E )1/-

and i > 1, then there exists an optimal solution in which jobs (i — 1, f) and (i, f) are scheduled

contiguously. Thus, composite jobs are formed from such pairs of jobs.

3 Lower bounds

In this section, we derive a new lower bound based on Lagrangean relaxation. Two alternative

methods of determining values of the multipliers using a multiplier adjustment technique and

if = the completion time of job (i, f) E Arl f). Also,

subgradient optimization are also described.

Based o n ideFas} 1 oaf n Fidfi saTh se=er t {[-4u1]p,. wfoer , g f7aiv}me. i Blay yt if md oeecf-icinunirdnseg x ivnea d[r ti fa—obr lm1e,s ut]l,ation of t(hie, fp)r oEb Alerm, .t LEe Tt T, denote

an upper bound on the completion time of the last job; for instance, T = E( , f)EN-(sf

let .F — {1 ,

Yft = t 0 otherwise, E t E T,

C

0 othXe,f t r= w{ 1 iif sjobe (i,, f) is processed in the interval [t — 1. t],

f

5

we obtain the following formulation:

minimize E WifCi f

EE,.fT)E r.A irf rt i-ft Pi ffEFyft 1 t( i,E f )T E JV ((32))

subject to

t

Eft— oft h\_e1rw- iXse2-1,f,t-1) ECiYf}ft' (t' f) E Ar (5)

(

onstraints (2) ensure that the requ ired+ proct-e1ssing of all jobs is ext eEc uTt.ed, while the mach(i8n)e

Sf(Xift - if t E {Cif - (i, f) E t = sf 1,... ,T (4)

e=t-sf

\_ 1

e

0

Cif > C2 -1,1 + (i, f) E N (6)

CO f 5 0 f E (7)

Y ft E {0,1} f E

C

capacity constraints are represented by (3). Constraints (4) ensure that the necessary set-ups are

performed: if job (i, f) is processed in the interval [t — 1, t], but neither job (i, f) nor job (i — 1, f)

is processed in [t — 2, t — 1], then job (i, f) is the first job in a batch and starts its processing

at time t — 1. Consequently, a set-up for family f is performed throughout the previous sf time

intervals. Constraints (5) ensure that the variables XI:ft are consistent with the completion time

vaeri anboltees t hCaitf ,t haen dL athgarat nngoe panre permopbtlieomn dise caollmowpoedse. sT; hthee SreW isP Ta soerpdaerrainteg swubitphrionb flaemmi lfioers e(aTchhe ofarmemil y1.)

is represented by constraints (6).

To obtain a lower bound, we perform a Lagrangean relaxation of the machine capacity con-

straints (3). If = „uT) is a given vector of non-negative multipliers, then we obtain the

lower bound

Lu(byj)e c=t mto i(n2{), E(4), (W5)i,f (C6i)f, +(7 E) an(d E(8). f EgEr kitY ft)} — tEET itt

tET (i,flEAr

s

W

In any subproblem, a cost it t is associated with performing processing or a set-up in the time

interval [t — 1, t].

To solve the Lagrangean subproblem for family f (f ...,F), we propose a dynamic pro-

gramming algorithm. We define a recursion on z(j,t) which represents the cost of scheduling jobs

(1, f),... , (j, f), where job (j, f) is completed at time t and t > s f -FELi Af . All initial values of

6 T using the recursion

z(j. t) are set to De. W e cto-Sm f p-1u0t1ef +z1(j, t) for t = sf + EL, pi f for j = 1,

z(j,t) = wi ft + E+1

=

or j = 1, we nmotine {t hza(jt —z( j1,,t t) — is pti{h fze)( wj+ —e wi g1j hf,t tet +'d)} c o+m VIp flet tion =tti-ms ef forf +jo1bt t(vj}, ff)o wr hje =n i2t ,c o..m. ,p nlef.tes at

f

min

f

C

he minim izat, ino1n, ctohnes cidoenrt rthibeu ctaiosens f tohra jto jbobs s( 1(,j f—)..1..,,(j)j —an d1 ,(fj),f m) aurset abses iagdndeedd t:o t hthee t wsaom tee rbmatsc ihn,

F

time t plus the sum of multipliers for the time periods during which (j, f) is processed. However,

for j — 2

t

and are scheduled in different batches, respectively. The minimum total cost for jobs of family f is

B(p) = E Kj —pi fE , p,tT. . Thus, we obtain the lower bound

K j = min{z(ni,t)},

whhee rceo tmhpe umtaitnioimn iozfa tLioBn( /i.s2 )o vreeqr uviarelus eOs (tN=ST)f t+ime. tET

L

fE-F

T

Ideally, we would like to choose p so that LB(/2) is as large as possible. We propose two

alternative approaches for the determination of /2. In the first, the multiplier adjustment method is

used. CT(hBis, )i s— a c1ot.+n A1stfr=tue( crt isveett pinrgoc edu=re 9 Efomr ceoAmrp tuv,t fi n—g m1/BuWl2t,iP pTli(e)BBrsr, ))f,r, owwmhh eearr ehe e WBur,Pi sT(tvi(cB =s,c) 1hies,. dc.u.o l,me .v pF)u iirtses tda,

wrhoeim sa pc(po1)nly,s o ttrthuhece trhi omenuu rylitisieptlildcie sor psfi t ta t h=ree 0cno efmoxrtp tuse tec=dt ir oe1nc ut,foro.s .rif vi.tne =dl,yT T a.u„ ssN-icFnoh gte1e,d tuhlae,t T,S s— ifno 1cr.e w {hAitc}h i sth neo jno-binsc wreivat sh—iinng 1e iaannc tdh,

family are sequenced in SWPT order. Let S =

batch which completes at time C(B,). Also, let To = 0, T, = C(B,) for v =

=

— 1/WPT(B,) for v = 1, , v. t = + 1, • • • 'Tv,

f

{

µ

/

T

machine time is more expensive for earlier time periods than for later ones. This pricing structure

is consistent with the observation that since more jobs are competing for machine time early in

the schedule, such time periods should be assigned a higher price.

W1e, .n..o,wF ,p rtohvenid LeB a( /t.h/)e =or eE,tBifceFasl)- ,tjv uwis fthCiefi(rceBa Bft)i1o. nis fao rb tahtcish c fhoorimcee do ff rmomul taip sliinergsl eb yjo sbh oowf fianmg itlhya tf tfhoer

lower bound is exact when every family contains a single job.

Theorem 3 If S = (B1

f

Proof. When each family contains a single job, the heuristic method of the next section generates

a schedule S in which batches are ordered in non-decreasing order of WPT values. Thus, we assume

without loss of generality that S = (B1 ,... , BF), where WPT(B1 ) < < WPT(Bp).

Let C ft = weft + E't,,t\_s, f+i pt, denote the cost in the Lagrangean problem of scheduling

jo(bB (F1.) f—) 1to, wcoem obptlaeitne farto tmim (e9 )t ,t hfoaCrt ftfth e r=eW ilsf sFom(11 e4 af-5na1md-P it1l y =f +fs1' ,f- wPft-h I-e•1r) •e• f .' E;T {. fW,..e., Fsh}ow, f fo irr swt ht(h1ica0<ht)

Cft < Cf,t+i for t = C(B f),... ,T — 1. Clearly,

rom the definition of WP T(Bf) i-n (1) ,= w W0e ofbolftra it(n' P =th iCaft (tSBh feF) Ir)Wi,g.P.h.T,tT -(h1 a— 3n Jd )1 .s .i dAel osof ,( 1fo1r) tis' e=q uCa(l Bto.1 )z• e(• 1r•o1,)

Cf,t+1

From (9), we observe that

C

— = 1/WPT(Bp): the indexing of batches using WPT values implies that itt, —

1/WPT(Bf). We have now established that tit , — f.te.4. 1 < 1/WPT(Bf ) for t' = C(B f),... ,T — 1.

Substituting in (10). we obtain

Cf,t+1.- Cft

iete n+o1w = s h1o/Ww PthTa(t Bcf1t, )> > c 1f ,/tW+i PfoTr( Bt =f) .s fTf he,re.fo.r e,• , C (B Cf)( B— 1f). ,A...n,Tal o—go u1.sly with fo }u,r f porre wvihoicuhs

F

hich yields cft > cf,t+1 for t = S f f , C(B f) - 1, as required.

which establishes the desired inequality C ft < cf,t+i for t

W

analysis, for t' = sf Pl .?' • • • , C(CBf tf )5 - 1W,l thfe re( Pis is.ofm sef) /faWmPilTy (Bf'f,) ,w--h--e r0e, f' E

—

Cf,t+1

w

The above analysis shows that there exists an optimal solution of the Lagrangean problem

icno nwsthriacinht se a(3c)h ibs astactihs fBie1d iws isthc heeqduuallietdy itno ac osomluptlieotne oaft tthime Le aCgr(Ban fg)e. anT phruobs,l eemac. hT hoifs tehseta rbelliashxeesd

that LBW = Efe wi fC(B f). q

The tightness of the lower bound that we obtain when the multiplier adjustment is used to find

p, is dependent on the quality of the schedule S. Initially, S is obtained by applying the heuristic

method that is presented in the next section. However, whenever a better solution is found in the

branch and bound algorithm, S is updated and multipliers are computed from this new schedule.

Thus, the multipliers change only when an improved upper bound is found.

The second approach for determining values of the multipliers uses subgradient optimization.

Let / (1) be a vector of multipliers at iteration 1 of this procedure, and let e) denote the number

of families for which a set-up or some processing of a job is scheduled in the interval [t — 1, t] in

the computation of LB(A(1) ). Initially, we set p(1) to be the multipliers obtained by the multiplier

adjustment method, wlhite= re m S ias xob{tapini(eI)d +by (f1i)r(sUt fBo r—m LinBg( faL s(1in))g)l(ee) b —atc 1h) from each family and then

sequencing these batches in non-decreasing order of their WPT values. Thereafter, we set

(1+1) 1'

A

ET-i(e) - 1)2

8where UB is an upper bound on the minimum value of the total weighted completion time and

A( I ) is a scalar step length which satisfies 0 < A(1) < 2. At the root node of the branch and bound

search tree. we set A(1) = 2, and perform 20 subgradient optimization iterations, halving the step

length when 5 successive iterations fail to improve the lower bound. For other nodes of the search

tree, we set ,a(1) to be the vector of multipliers at the parent node, we set A (1) = 0.1, and perform

3 subgradient optimization iterations, halving the step length when no improvement in the lower

bound is observed for 2 successive iterations.

4 A heuristic method

In this section, we propose a heuristic method which is applied at the root node of the branch

and bound search tree to find an upper bound UB on the minimum value of the total weighted

completion time. Our heuristic uses a constructive procedure, which is based on observations of

Gupta [6], to find an initial schedule. Descent procedures are then applied with the aim of reducing

the total weighted completion time. A descent method attempts to improve on a current solution by

searching, in some suitably defined neighbourhood, for a new solution which has a lower objective

function value. If such an improved solution is found, it becomes the current solution from which

further improvement is sought. If no improvement is possible, then the method terminates and the

solution is a local optimum.

Heuristic method

(3 f and mf = 1

Step 1. Index the jobs within each family in SWPT order. Set of =

for f = 1.... , F. Set k = 1 and g = 0 (where k is the first unfilled position in the sequence

and g is the family to which the last scheduled job belongs).

Step 2. Choose family f such that af is as small as possible. Schedule job (mf, f) in position k,

set mf = mf + 1 and k=k+ 1. If mf > nf , set af = co; otherwise, set cif =

If g = 0, set g = f . If g f, set ag = (sgwm9s and g = f.

Step 3. If k < N, then go to Step 2.

Step 4. Apply shift-batch: a descent algorithm which uses a neighbourhood based on the inter-

change of adjacent pairs of batches.

Step 5. Apply shift-job: a descent algorithm where the elements of the neighbourhood are con-

structed as follows. For each batch, its last job is shifted backwards to the start of the next

batch containing jobs of the same family (unless it is the last batch of a family), or its first

job is shifted forwards to the end of previous batch containing jobs of the same family (unless

it is the first batch of a family). The last job of the last batch of a family is also shifted

backwards, thus creating a new batch.

Step 6. Apply shift-batch.

The schedule generated by the heuristic method is consistent with Theorem 1 as it has the

property that the jobs within each family are sequenced in SWPT order.

95 Branch and bound algorithms

In this section, we describe three branch and bound algorithms. The first is based on forward

branching with objective splitting (FBOS), and is essentially the algorithm of Mason and Ander-

son [10]. At the root node of the branch and bound search tree, we apply the preprocessing routine

that is described in Section 2 to form composite jobs and thereby reduce the problem size. Also,

the heuristic method of Section 4 is applied to generate an initial upper bound. At each node of the

branch and bound search tree, Rules 1 to 5 of Section 2 are used as pruning devices (Rule 3 is not

used in the original algorithm of Mason and Anderson), together with the dynamic programming

dominance rule. A depth-first search strategy is used, where the immediate descendants of a node

are explored in non-decreasing order of their lower bounds.

The second branch and bound algorithm employs forward branching and multiplier adjustment

(FBMA). Apart from the method of computing lower bounds, it is identical to the first algorithm.

Initially, the lower bound LB(p) uses multipliers that are obtained from the heuristic schedule.

Within the computation of the lower bound, an upper bound is also computed: form batches from

the jobs of the same family that are scheduled contiguously in the lower bounding computation,

use Theorem 2 to sequence these batches, and evaluate the resulting schedule. If the resulting

upper bound is less than the current upper bound, then UB is updated, and multipliers are based

on this improved schedule in subsequent lower bounding computations.

Our third branch and bound algorithm uses a binary branching rule and subgradient optimiza-

tion (BBSO). At the root node, the preprocessing routine of Section 2 is applied. At each node

of the search tree, we compute an upper bound by constructing two heuristic schedules. The first

heuristic forms a batch from each family and then sequences these batches in non-decreasing order

of their WPT values. In the second heuristic, jobs are sequenced in SWPT order. The Lagrangean

lower bound is computed at each node, as described in Section 3, where subgradient optimization is

used to find values of the multipliers. Our branching rule selects a pair of jobs (j – 1, f) and (j, f),

where (j, f) E N and j > 1, and constrains these jobs either to be contained in the same batch,

or to be contained in different batches, Unlike forward branching, this binary branching allows the

possibility of making important decisions at the higher levels of the branch and bound search tree.

For the branch in which jobs (j – 1, f) and (j, f) must be contained in the same batch, we form

a composite job from (j – 1, f) and (j, f). In the other branch in which job (j, f) starts a new

batch, we partition the jobs of family f into two families containing jobs (1, f),... ,(j – 1, f) and

jobs (j, f),... , (n f , f), respectively. We select job (j, f) so that the Lagrangean problem sched-

ules jobs (j – 1,j) and (j, f) to complete at time t' and t, respectively, where t' p3 f < t and

z(j, t' p3 z(j, t) is as large as possible (z(j,t' +pi f) – z(j, t) is obtained from the dynamic pro-

gramming recursion for the Lagrangean problem and provides a measure of the cost of not splitting

family f so that job (j, f) starts a new batch). For this binary branching rule, initial experiments

with our Lagrangean lower bounds, where multipliers are obtained using subgradient optimization

and using the multiplier adjustment method, indicate that the subgradient optimization approach

is superior. As in the two other algorithms, a depth-first search strategy is used.

106 Computational experience

In this section, we report on the results of computational tests to assess the effectiveness of the

three branch and bound algorithms. Algorithms FBOS and FBMA were coded in ANSI-C, whereas

BBSO was coded in FORTRAN 77. All algorithms were run on a HP 9000/715 computer. For

FBOS. computation is abandoned if a limit of 50 000 nodes is exceeded, whereas a time limit of

600 seconds is used for BBSO. No limits are applied for FBMA (all problems can be solved using

reasonable computational resources).

Test problems with 30, 40 and 50 jobs, and with 4, 6, 8 and 10 families were generated as

follows. For each combination of N and F, the jobs are uniformly distributed across families, so

that each family contains either [N/FJ or [N/Fl jobs. Processing times and weights are randomly

generated integers from the uniform distribution defined on [1,10]. Since the size of set-up times

relative to processing times may affect problem 'hardness', we generated problems with small (S),

medium (M) and large (L) set-up times. Medium set-up times are randomly generated integers

from the uniform distribution defined on [1,10]. Having generated an instance with medium set-up

times ,s! (f = 1, , F), corresponding instances with small set-up times rs f /21 and with large

set-up times 2sf were constructed. For each of the 12 combinations of N and F, 50 test problems

with small, medium and with large set-up times were created.

Our test problems cover a range of different scenarios. The number of jobs per family ranges

from 3 (for N = 30 and F = 10) to 12 and 13 (when N = 50 and F = 4). Also, a variety of set-up

time ranges is considered. Problems with very small set-up times are likely to exert little influence,

so that an optimal sequence of jobs will be close to SWPT. On the other hand, batches will be

formed from complete families when set-up times are very large. Both of these extreme cases are

likely to be relatively easy to solve. Possible schedules in our instances may involve the machine

undergoing a setup time for only about 4% of the time (for N = 50 and F = 4 when set-up times

are small and there is only one set-up for each family), or for about 67% of the time (for large

set-up times when a set-up time is incurred for every job).

Table 1 gives computational results for the three branch and bound algorithms. For each

combination of N and F. average computation times in seconds (ACT), average numbers of nodes

(ANN) and numbers of unsolved problems (NU) are listed for the instances with small, medium

and large set-up times. When there are unsolved problems, the values listed under ACT and ANN

are lower bounds on the true averages.

We first observe from Table 1 that there are significant numbers of unsolved problems for

algorithms FBOS and BBSO. However, FBMA solves all test problems without generating large

search trees or requiring excessive computation time. For instance, among the 600 test problems

with medium set-up times, only 6 require more than 3 minutes of computation time.

The computational effort varies significantly with F for algorithms FBOS and BBSO, but is

fairly stable for FBMA. Algorithm FBOS captures some of the problem structure through its

dominance rules. thus enabling it to solve the problems with a small number of families. However,

the objective splitting bound is too weak to control the explosion in tree size for larger problems,

especially when there are many families. Algorithm BBSO also captures part of the problem

structure by its use of a binary branching rule that splits the 'right' family. This algorithm is clearly

1 1Table 1. C omparative computational results.

Set-up FBOS FBMA BBSO

times N F ACT ANN NU ACT ANN ACT ANN NU

S 30 4 0.1 657 - 1.9 74 24.6 440 -

6 0.4 1775 - 2.7 92 9.1 113 -

8 2.0 3732 - 2.7 95 9.0 103 -

10 1.0 2778 - 1.6 66 5.5 53 -

40 4 0.7 2477 - 11.2 184 211.1 2401 7

6 7.5 9096 - 11.8 161 119.8 1127 5

8 41.0 22167 7 12.0 171 60.4 507 1

10 37.0 20292 7 9.5 148 34.6 277 1

50 4 3.4 5591 - 34.5 365 448.4 3554 34

6 68.5 29478 9 56.0 437 305.9 2124 17

8 112.0 44658 35 69.0 451 227.8 1334 9

10 112.0 46818 42 61.0 436 160.4 891 7

M 30 4 0.1 371 - 1.6 44 7.9 108 -

6 0.2 757 - 1.9 53 5.5 64 -

8 0.3 1199 - 1.4 43 3.3 28 -

10 0.3 956 - 1.3 45 3.2 26 -

40 4 0.4 1522 - 11.3 165 119.7 1243 5

6 1.9 4082 - 10.0 112 47.6 401 1

8 10.5 8703 1 11.6 143 37.8 298 -

10 6.0 6992 - 7.1 94 13.2 85 -

50 4 1.4 3456 - 31.0 217 331.0 2528 2

6 23.5 15226 - 46.5 263 149.9 937 3

8 67.0 29040 18 54.0 281 109.6 593 3

10 79.0 33810 24 32.0 200 92.2 485 1

L 30 4 0.1 198 - 1.4 30 5.5 69 -

6 0.1 294 - 1.0 22 2.2 19 -

8 0.1 322 - 0.8 22 2.1 16 -

10 0.1 257 - 0.8 21 2.0 14 -

40 4 0.1 661 - 7.8 65 38.4 354 1

6 0.3 1292 - 5.8 51 13.2 95 -

8 0.4 1655 - 4.8 49 12.1 83 -

10 0.6 1770 - 2.8 34 5.5 29 -

50 4 0.4 1535 - 24.7 113 126.6 845 5

6 4.1 5215 - 31.5 180 103.7 620 3

8 10.7 8225 2 23.5 118 18.2 88 -

10 18.3 11633 2 18.1 101 23.4 105 -

FBOS: algorithm with forward branching and objective splitting.

FBMA: algorithm with forward branching and multiplier adjustment.

BBSO: algorithm with binary branching and subgradient optimization.

ACT: average computation time in seconds on a HP 9000/715 computer.

ANN: average number of nodes in the branch and bound search tree.

NU: number of unsolved problems (out of a total of 50).

12quite effective when there are many small families. With few large families, however, there are many

ways to split each family, and the Lagrangean bound in combination with subgradient optimization

is not strong enough to restrict the size of the search tree. The disadvantage of BBSO is the fact

that it fails to take advantage of the problem structure as embedded in the dominance rules.

Algorithm FBMA exhibits a stable behaviour. It combines the efficient exploitation of dominance

rules within the search tree with a quick implementation of a reasonably good Lagrangean bound.

The result is that all problems can be solved quite effectively, and the algorithm is not very sensitive

to the relative size of N and F. Dominance Rule 3 is particularly effective for problems with larger

numbers of families. As an illustration, for the original implementation of algorithm FBOS by

Mason and Anderson [10], which does not use Rule 3, problems with F = 8 and F = 10 are

much more demanding on computer resources compared with our version which includes Rule 3.

Without Rule 3 in algorithm FBOS, problems with F = 10 are harder to solve than those with

F = 8. The use of Rule 3 helps to overcome the inability of the objective splitting lower bound to

restrict the search for large F.

As expected, the relative size of set-up times affects problem hardness. Results in Table 1

indicate that all algorithms find the test problems with small set-up times to be the hardest, and

those with large set-up times are the easiest. With large set-up times, family splitting is very

expensive. As a result, the splitting of families into batches in an optimal solution is limited, and

the combinatorial characteristics of the problem are reduced. As set-up times become smaller,

more family splits become potentially attractive, and the problem achieves its full combinatorial

complexity. Algorithms FBOS and BBSO experience difficulty in coping with this complexity in

our test problems, although FBMA performs adequately.

In terms of average computation times, algorithm FBOS dominates BBSO and FBMA for

problems with a few large families. Its 'quick and dirty' lower bound is more efficient than the

more time consuming Lagrangean method for these easier problems. For the harder problems,

algorithm FBMA is clearly superior to both FBOS and BBSO.

Having established that algorithm FBMA is superior to the other branch and bound algorithms,

we report on the results of some further tests with FBMA. We use test problems with 60 and 70

jobs and with 4, 8 and 15 families that are generated in a similar way to the other problems. These

additional tests were run on a HP9000/G50 computer, which is about twice as fast as the HP

9000/715 on which the previous results are obtained. A time limit of 300 seconds for algorithm

FBMA is used. Table 2 shows that algorithm FBMA becomes rather time consuming for 60- and

70-job problems. This is mainly due to the dynamic programming recursion within the lower bound

computation. The table also gives an indication of the number of batches in the optimal solution,

i.e., of the number of times families are split on average.

7 Concluding remarks

This paper considers the problem of scheduling families of jobs on a single machine to minimize the

total weighted completion time, where a set-up time is incurred whenever the machine switches

from processing a job in one family to a job in another family. A new lower bounding scheme

13

Table 2. Computational results for FBMA on larger problems.

Set-up FBMA No. of batches

times N F ACT ANN NU Min Max

S 60 4 41.7 429 - 10 15

8 97.6 715 3 16 25

15 67.2 645 2 24 32

70 4 93.4 611 1 11 17

8 175.7 976 15 18 26

15 144.4 811 8 27 36

M 60 4 35.0 268 - 8 14

8 61.6 405 1 14 22

15 28.0 296 - 22 29

70 4 82.5 399 1 10 14

8 149.4 637 8 15 23

15 107.1 535 8 24 30

L 60 4 32.9 163 - 7 11

8 33.6 173 - 13 18

15 10.3 82 - 17 25

70 4 70.8 224 - 8 11

8 108.1 361 6 13 20

15 37.9 148 - 20 27

FBMA: algorithm with forward branching and multiplier adjustment.

ACT: average computation time in seconds on a HP 9000/G50 computer.

ANN: average number of nodes in the branch and bound search tree.

NU: number of unsolved problems (out of a total of 50).

14based on a Lagrangean relaxation of machine capacity constraints is derived. Also, a multiplier

adjustment method for obtaining quickly computed values of the multipliers is proposed.

Computational results show that when the Lagrangean multiplier adjustment bound is used in

a branch and bound algorithm which adopts a forward branching rule and incorporates various

dominance rules (algorithm FBMA), problems with 70 jobs can be solved using reasonable com-

putational resources. This algorithm is far superior to the branch and bound algorithm of Mason

and Anderson [10] (algorithm FBOS) which uses objective splitting in its lower bounding scheme.

Computational tests are also reported for another algorithm which uses a Lagrangean subgradient

optimization bound and which employs a more flexible branching rule (algorithm BBSO). The dominance rules cannot be used with this branching rule. Results indicate that the increased flexibility offered by the branching rule does not compensate for extra computation that is incurred due to the inapplicability of the dominance rules.

Our computational results provide further evidence to support the view that, in a Lagrangean bounding scheme, a multiplier adjustment method for determining values of the multipliers is often preferable to the computationally expensive subgradient optimization method.

Scheduling problems which involve batching have both theoretical interest and practical importance. Studies which develop and test algorithms for solving such problems are relatively scarce. However, for minimizing total weighted completion time on a single machine, this paper, together with the study of Crauwels et al. [3] which develops various local search heuristics, provides a thorough treatment of the problem. An important area for future research is to design enumerative algorithms and local search heuristics for single machine problems with other objectives, and to consider multi-machine models.

Acknowledgements

This research was partly supported by NATO Collaborative Research Grant 0224/88 and by the International Association for the Promotion of Cooperation with Scientists from the Independent States of the Former Soviet Union, INTAS-93-257.

Ветвей и границ АЛГОРИТМЫ  
SINGLE МАШИНА диспетчеризацию  
BATCH SET-UP ВРЕМЯ, ЧТОБЫ УМЕНЬШИТЬ TOTAL  
WEIGHTED ВРЕМЯ ЗАВЕРШЕНИЯ  
от  
H.A.J. CRAUWELS \*  
A.M.A ХАРИРИ \*\*  
C.N. POTTSt  
а также  
Л.Н. VAN WASSENHOVEtt  
95/70 / TM  
  
Ветвей и границ Алгоритмы одной машине  
Планирование с Batch время настройки, чтобы минимизировать  
Общий взвешенный Время завершения  
H. A. J. Crauwels  
KIHDN  
J. De Nayerlaan 5  
2860 Sint-Katelijne-Waver, Бельгия  
А. М. А. Харири  
Департамент статистики  
Король Абдул Азиз университет  
Джидда 21413, Саудовская Аравия  
С. Н. Поттс  
Факультет математических исследований  
Университет Саутгемптона  
Саутгемптон 5017 IBJ, Соединенное Королевство  
Л. Н. Ван Wassenhove  
INSEAD  
Бульвар-де-Констанс  
77305 Фонтенбло, Франция  
Июля 1995  
Абстрактные  
В данной статье представлены несколько ветвей и границ алгоритмов для одной машины планирования  
Проблема с дозированием. Работа разбиты на семьи, и время настройки необходимо, когда  
есть переход от обработки заданий одной семьи на работу в другой семье. Цель состоит в том, чтобы  
минимизировать общее взвешенное время завершения. Нижняя граница на основе лагранжева релаксации  
этого ограничения мощности машины происходит. Кроме того, метод корректировки множитель, чтобы найти  
Предлагается значения множителей. Вычислительный опыт работы с экземплярами, имеющих до 70  
рабочих мест показывает, что нижние границы являются эффективными в ограничении поиска.  
1. Введение  
Многие практические проблемы планирования предполагают обработку нескольких семейств связанных заданий на общих  
объекты, где время установка наступает всякий раз, когда есть переход от обработки задания в  
одна семья на работу в другой семье. Чтобы избежать установки окна, несколько рабочих мест одной семьи может быть  
запланирован смежно, чтобы сформировать партию. График определяет, как формируются партии, и определяет  
порядок обработки партий.  
1As пример задачи, которая включает в себя планирование и пакетирование, Ahn и Hyun [1] описывают  
применение при изготовлении стальных труб. Процесс производства труб требует  
использование прокатной машины, в котором ролик, который будет использоваться, зависит от наружного диаметра трубы.  
изменение ролика соответствует подстроено. Таким образом, заказы для труб с одинаковым наружным диаметром  
лежат в пределах одной семьи рабочих мест в нашей модели. Анализ алгоритмов и сложности для различных  
Проблемы планирования, которые включают пакетирование дается Поттс и Ван Wassenhove [14].  
В этой статье мы рассмотрим одну задачу планирования машина с N рабочих мест, которые являются части-  
упоминавшейся в F семей. Время настройки являются последовательность независимыми: каждый из них зависит только от семьи  
задания должны быть обработаны в следующем. Мы предполагаем, что все рабочие места доступны в нулевой момент времени и что  
каждая работа имеет заданное время обработки и связанный с ним положительный вес. Мы хотим найти расписание  
что сводит к минимуму общее время взвешенная завершения выполнения заданий.  
Когда все время наладки равны нулю, то задача решается в 0 (N) N войти время секвенированием  
рабочих мест в кратчайшие взвешенное время обработки (SWPT) заказ (без порядке убывания времени обработки  
взвешивать соотношение). Monma и Поттс [11] показывают, что для ненулевых времени наладки, работы в рамках каждого  
семья упорядочиваются в SWPT порядке в оптимальном расписании. Используя это свойство, они получают  
динамический алгоритм программирования, который требует O (F2N2F) время. Более эффективный динамический про-  
Алгоритмы граммирования были разработаны впоследствии. Для случая единичных весов, Ahn и  
Hyun [1] разработать алгоритм, который требует 0 (F2NF) время, и Гоша [5] предлагает в O (F2NF)  
Алгоритм для произвольных весов. Таким образом, при фиксированном F, проблема полиномиальное, хотя  
алгоритмы обеспечивают практический способ решения только тогда, когда F очень мала. Вопрос о  
является ли проблема NP-трудной для любого F является нерешенным, даже для случая единичных весов.  
Эвристические методы также являются предметом некоторых исследований. Для случая единичных весов, Гупта [6]  
Девизес эвристический SPT основе. По сути, это однопроходный жадные процедура, которая требует  
0 (N) N войти время: задание будет обрабатываться следующим выбрана таким образом, что она имеет наименьший пополнений  
время среди всех ние рабочих мест кандидатов. Несмотря на то, имея преимущество скромных вычислительных  
требования, качество графики, которые он генерирует часто довольно бедным.  
Различные местные поисковые эвристика также предложены в литературе. Ahn и Hyun [1] разработать  
Спуск (или итеративный улучшение) эвристический. Они используют окрестность, в которой блоки рабочих мест  
та же партия смещаются от одной части к другой последовательности, но с тем ограничением, что  
SWPT упорядочение рабочих мест в семье сохраняется. Мейсон [9] предложен генетический алгоритм  
которая использует наблюдение, что знание которых начинается работа каждой партии позволяет решение  
быть построена путем упорядочения партий, используя обобщение правила SWPT. Таким образом, решения  
могут быть представлены в виде двоичных строк, к которым могут быть применены стандартные генетические операторы. Crauwels  
и другие. [3] разработать моделируемый отжиг, порог приема и методы поиска Табу, которые строят  
о работе Ahn и Хен. В вычислительных тестах, Crauwels и др. найти свой поиск с запретами  
алгоритм превосходит многозаходной версии метода спуска по Ahn и Hyun а, чтобы  
моделируемый отжиг и порог приема, а также генетического алгоритма Мейсона.  
Мейсон и Андерсон [10] предложена ветвей и границ. Особенностью их  
Алгоритм является широкое использование правил доминирования, чтобы ограничить размер ветвей и границ поиска  
дерево. Они используют правило ветвления вперед, которое дает частичные графики, в которых виртуализированных рабочих мест  
в начальных позициях. Их нижняя граница получается с помощью объективного расщепления: общее взвешенное  
2completion время можно разбить на вклады от времени обработки и от  
раз установка, оптимизированные по отдельности. Правило SWPT минимизирует первый взнос.  
обобщенное правило SWPT применяется к семьям, где время обработки семьи заменяется его  
время настройки, сводит к минимуму второй. Численные результаты указывают на то, что этот алгоритм представляет  
практический способ решения при условии, что не более чем около 30 рабочих мест. ограничение  
в сравнительно небольших случаях объясняется слабостью нижней границы: правила доминирования  
основной вклад в обрезку ветвей и границ дерева поиска.  
Несмотря на значительное недавнее научно-исследовательской деятельности в области моделей планирования в которых участвуют  
пакетирование, было предпринято несколько попыток спроектировать вычислительно эффективные алгоритмы перечислительные.  
Это печально с точки зрения практической значимости этих проблем. В этой статье мы предлагаем  
новая ветвей и границ алгоритм, который превосходит, что Мейсона и Андерсона [10]. Оно использует  
нижние границы, которые вытекают из лагранжевой релаксации ограничений машины мощности.  
Особенностью является использование метода корректировки множителя для получения значений мультипликаторов.  
Возможность построения умножителей с некоторыми желаемыми свойствами укрепляет мнение, что часто  
можно использовать структуру проблемы в лагранжевом процедуры, а не прибегать к большегрузных  
передал оптимизации субградиент подход [2,7,12,13]. Наши результаты расчетов показывают,  
что качество нижних оценок, полученных из процедуры настройки умножитель высока.  
В разделе 2 мы даем официальное заявление нашей проблемы и представить некоторые доминирование крите-  
RIA. Раздел 3 вытекает наша нижняя грань используя Лагранжа релаксации емкости машины  
ограничения. Наша процедура регулировки мультипликатор для нахождения значений мультипликаторов также пре-  
ставлены. Раздел 4 дает эвристический метод для планирования заданий. Ветвей и границ алгоритмы  
описаны в разделе 5. В разделе 6 докладов по вычислительным опытом с филиалом и  
связанные алгоритмы и некоторые заключительные замечания приведены в разделе 7.  
2 структура и доминирование правила Проблема  
Чтобы сформулировать нашу проблему планирования с установленным планом раз пакетными более точно, нам дано множество .A1  
содержащий N рабочих мест, которые разделены на F семей. Каждая семья F, для F = 1, Ф. содержит п  
рабочих мест. Все рабочие места становятся доступными для обработки в нулевой момент времени и должны быть запланированы на один  
mmamcheidniea.t eFlyo ра tfhteer ia't Hjo JBO б в о аф d <faipfmf, eifrl, eyfn 1ftw, е а, WFM, hfii lcfyho. . RWA оскверненного s = од е, 1 па, на т айн bitFyia (Li, s EFT) -, у LPE тт IPMI Fe DS Fe NISO дерево qitusi rperdo CIFE SAS jionbg ftrimome  
и его WIF положительным весом. Мы предполагаем, что рабочие места в пределах каждой семьи индексируются в SWPT  
заказ. Таким образом, р е / WIF <  
Последовательность автономной наладки Время с е> 0 наступает всякий раз, когда работа в семье F обрабатывается  
я  
Семейство F сначала обрабатывается на станке. Таким образом, обработка каждой партии должна предшествовать  
соответствующей установки.  
Сначала приведем две фундаментальные результаты о структуре оптимального расписания.  
Теорема 1 (Monma и Поттс [11]). В любом оптимальном расписании, рабочие места в пределах каждой семьи являются se-  
quenced в SWPT порядке.  
Из теоремы 1, то задача сводится к одному из слияния SWPT упорядоченных списков заданий для  
разные семьи, чтобы сформировать последовательность.  
3 Рассмотрим любой график S = (B1,., В ") ???, где В, (v = v) является периодическим, содержащей Максвеллом  
IMAL последовательных подпоследовательности рабочих мест из того же семейства Р ". Определим взвешенную обработку  
timhee е (oWlloPwTi) п Гра rteiosu Flot гк eivaecsh AB gaetcnhe rBal, я zteod б SeWPTi, RFU) EleB ФЖ или партии (ie.fs) EthB a.t использует WPT отношения. (1)  
WPT (Bo) = (8 е, Е р, е) I E  
(  
T  
Теорема 2 (Mason и Андерсон [10]). В любом оптимальном расписании, партии будут упорядочены в не-  
порядке убывания коэффициентов WPT.  
Теперь мы опишем некоторые правила доминирования, которые позволяют ограничить набор кандидатов  
Секвенирование следующего в продолжении некоторого начального частичного графика S = (B1, ..., B), где В,  
= 1, ..., v) является периодическим. Эти правила основаны на результатах Mason и Андерсона [10]. Мы  
Предположим, что S удовлетворяет условиям теоремы 1. Пусть м-ф обозначает число рабочих мест семьи  
F (F = 1, ..., F), которые содержатся в S, где 0 <М.Ф. <  
Предположим, что B, является партия некоторого семейства г, где г = е ", и эта работа (тк - (- 1, ч), где  
ч г, является кандидатом быть секвенировали сразу после последнего задания из S. Если тк 0, пусть работу (тк, ч)  
содержаться в пакетном Bu, где 1 <и <v и Н = е ,.  
Наши правила доминирования заключаются в следующем.  
Правило 1. Предположим, что мг <нг и WPT (Bo)> PMG + I, г / WRNG + я, г. Тогда работа (M9 + 1, г) должна быть  
запланировано сразу же после последней работы в S.  
Правило 2. Предположим, что V> 1, и WPT (Bv\_i)> WPT (БВ). Если мг = N9, то S не может служить  
начать оптимального расписания. Если мг <N9, то работу (мг + 1, г) должны быть запланированы сразу  
после последнего работу С.  
е Ei = фм, +1 пи е) I = Ehi им, +1} WIF. Если мг =  
Правило 3. Предположим, что WPT (B)> minfE {plmf, <, J} {(s  
нг, то S не может сформировать начало оптимального расписания. Если мг <нг, то работу (мг + 1, г)  
должны быть запланированы сразу после последней работы в S.  
Правило 4. Пусть мг <нг и р "" ± 1, h1w, Н, 4. 1, ч> PMG + 1,9 / WM9 + 1, г. Тогда нет оптимальной  
расписание, по которому работа (Mk + 1, ч) планируется сразу же после последней работы в S.  
Bu)> Prah + 1, h1 WMH + 1, ч, где v = и + я ??? I, F E) EB. Пиф)) / v. =. и. • +. • 0,1 • ВВО) Ео RW WiPfI.ABo-е-т,  
Правило 5. Пусть Mh> 0 и либо р ,,,, ч / Wm ,, ч>  
,  
Курица не существует оптимальное расписание я п whEich (jSolby (тк + 1, ч) планируется сразу же после того, как  
WPTABv + 1, ..., Bv) = (ш  
(  
T  
последняя работа С.  
4We в настоящее время оправдывает справедливость этих правил. Правила 2 и 3 следуют из теоремы 2. Под  
условия Правила 2, график S не согласуется с этим WPT упорядоченности, если дальнейшие задания не являются  
добавляют к партии В. Правило 3 считает, верхняя граница наименьшего значения WPT, которое может произойти  
когда формируются партии внеплановых заданий. Если WPT (B. ") Превышает это верхняя граница, далее  
рабочие места должны быть добавлены к партии В "в попытке удовлетворить WPT упорядочивание. Справедливость Правил  
1, 4 и 5 вытекают из следствия 4.1, следствия 4.2 и теоремы 4 Мейсона и Андерсона [10],  
соответственно.  
Правила 1. 2. 4 и 5 каждая используется Mason и Андерсона [10] в их ветвей и границ  
алгоритм. Тем не менее, правило 3 не используется в каких-либо предыдущих исследованиях.  
Динамическое правило программирования доминирование также полезно. Для двух начальных частичных расписаний Si и  
S2, которые имеют одинаковые конечные рабочие места и которые содержат одни и те же рабочие места, если 52 имеет время завершения  
его последней работы и общей взвешенной времени завершения, что нет меньше, чем соответствующая  
Значения для Si, то S2 доминирует S1.  
К сожалению, правила с 1 по 5 и динамическое правило программирования доминирование может быть использовано  
в дереве ветвей и границ поиска только тогда, когда переднее правило ветвления используется. попытки  
принимать важные решения в начале за счет использования гибкого правила ветвления должны быть сбалансированы  
от потери этих правил доминирования.  
Полезным процедура предварительной обработки предлагается в соответствии с Правилом 1. Предположим, что из> 2 и (ы F D-ПИФ) / WIF>  
P2F / W2 F для некоторого семейства F. Правило 1 гарантирует, что оптимальное расписание не содержит партию, содержащую  
только работа (1, е). Таким образом, рабочие места (1, е) и (2, е) должна быть запланирована смежно. Используя теорию  
е р2 и е  
Лоулера [8], они могут быть объединены, чтобы сформировать одну составную работу с временем обработки пи  
вес VII F W21. Хотя эта композиция увеличивает общее взвешенное время завершения через А  
постоянный член p21w11, приводит к эквивалентной задаче. После переиндексации заданий, этой предварительной обработки  
шаг повторно. Мейсон и Андерсон [10] показывают, что если = - р, fitllif, где (я, е) Е) 1 / -  
и я> 1, то существует оптимальное решение, в котором рабочие места (я - 1, е) и (I, F) запланированы  
смежно. Таким образом, составные задания формируются из таких пар рабочих мест.  
3 Нижние оценки  
В этом разделе мы получим новую нижнюю границу на основе лагранжева релаксации. Два альтернативных  
методы определения значений мультипликаторов с использованием метода изменения множителя и  
если = время завершения работы (я, е) Е Эрл е). Также,  
Оптимизация субградиент также описаны.  
На основании п о ideFas} 1 олух п Fidfi Сат се = эр т {[-4u1] р ,. wfoer, г} f7aiv меня. я Blay уг, если мкр oeecf-icinunirdnseg х ivnea г [г-ти фа ОБР lm1e, с ут] л, данию т (ГИЭ, Ф.П.) г OEB Alerm, .t LEE Tt T, обозначим  
верхняя граница времени окончания последнего задания; например, Т = Е (, е) EN- (SF  
Пусть .F - {1,  
YFT = Т 0 в противном случае, Е Т Е Т,  
С  
0 othXe, т е г = ш {1 IIF sjobe (я ,, е) обрабатывается в интервале [т - 1 т],  
е  
5 ???  
мы получаем следующую формулировку:  
минимизировать E WifCi п  
EE, .ft) E R.A ??? IRF ПАЭ-фут ??? Pi ffEFyft 1 т (я, Е е) Т Е СП ((32))  
при условии  
T  
Eft - ??? ??? часто h\_e1rw- iXse2-1, е, т-1) E ??? CiYf} фт '(т' е) E Ar (5)  
(  
onstraints (2) гарантировать, что Requ IRED + proct-e1ssing всех рабочих мест доб ЕЕС uTt.ed, в то время как маш (i8n) е  
Sf (Xift - если т Е {Cif - (я, е) Е т = SF 1, ..., T (4)  
е = т-SF  
\_ 1  
е  
0  
Cif> C2 -1,1 + (я, е) Е Н (6)  
CO е 5 0 е Е (7)  
Y футов Е {0,1} F E  
С  
ограниченные возможности представлены (3). Ограничения (4) обеспечить необходимый набор окна являются  
выполняется: если задание (я, е) обрабатывается в интервале [т - 1, т], но ни одна работа (я, е), ни работа (я - 1, е)  
обрабатывается [т - 2, т - 1], то работу (я, е) является первой работой в партии и начинает его обработку  
во время т - 1. Следовательно, установка для семейства F выполняется на протяжении всего предыдущего времени научная фантастика  
промежутки времени. Ограничения (5) гарантировать, что переменные XI: футы согласуются со временем завершения  
vaeri anboltees т hCaitf, т Хаен дл athgarat nngoe panre permopbtlieomn Dise caollmowpoedse. ST; hthe ??? е шнекового Та soerpdaerrainteg интернет-провайдера swubitphrionb flaemmi lfioers е (aTchhe ofarmemil y1.)  
представлена ​​ограничениями (6).  
Чтобы получить нижнюю границу, мы выполняем Лагранжа релаксации емкости машины со-  
ничения (3). Если = "иТ) является данный вектор неотрицательных множителей, то получим  
нижняя граница  
Лу (BYJ) ес = т МТО я (п2 {), Е (4), (W5) I, F (C6i) е, + (7 е) (d Е (8) F Эгер KITY фт.)} - Teet ITT  
Tet (я, Flear  
s  
W  
В любом подзадачи, стоимость его т связано с выполнением обработки или подстроено во времени  
интервал [т - 1, т].  
Для решения лагранжевых подзадачи для семейства F (F ..., F), мы предлагаем динамическую pro-  
граммирования алгоритм. Определим рекурсию на Z (J, T), который представляет собой стоимость работ планирования  
(1, е), ..., (J, F), где работа (у, е) завершается в момент времени т и т> s е -FELi М. Все начальные значения  
6 ??? T с помощью рекурсии  
г (к. т) установлены в De. W е CTO-Sm е р-1u0t1ef + z1 (J, T) при т = SF + EL, р е для J = 1,  
г (J, T) = Wi футов + E + 1  
знак равно  
или J = 1, мы nmotine {T HZA (JT -z (j1,, тт) - это PTI {ч FZE) (WJ + -e Wi g1j ХФ, т Tet + 'г)} м VIp FLET ние с о + = TTI-мс эф ??? + при \* jo1bt т (VJ}, ФФ) о сог hje = п i2t, с o..m. , Р nlef.tes в  
е  
мин  
е  
С  
он миним Изат, ino1n, ctohnes cidoenrt rthibeu ctaiosens F tohra JTo jbobs s (1 (, JF -) .. 1 .. ,, (J) J -an d1, (FJ), фм) aurset abses iagdndeedd т: OT hthee т wsaom тройник rbmatsc IHN,  
F  
Время т плюс сумма множителей для периодов времени, в течение которого (J, F) обрабатывается. Однако,  
для J - 2  
T  
и запланированы в разных партиях, соответственно. Минимальная общая стоимость работ семейства Р  
В (р) = Е ??? Kj -pi ??? fé , Р, Tt. , Таким образом, мы получим нижнюю границу  
K J = тт {г (п, т)},  
whhee rceo tmhpe umtaitnioimn iozfa tLioBn (/i.s2) о vreeqr uviarelus ЭОС (Tn = ST) F T + IME. ??? Tet  
L  
fé-F  
T  
В идеале, мы хотели бы выбрать р так, что LB (/ 2) как можно больше. Мы предлагаем два  
альтернативные подходы к определению / 2. В первом случае, метод корректировки множитель  
используемый. КТ (hBis,) в c1ot это-. + П = A1stfr Вт (элт isveett pinrgoc Edu = ге 9 Efomr ceoAmrp TUV, т фи н-г м1 / BuWl2t, Ip pTli (е) BBrsr,)) Р, owwmhh eearr ехе е WBUR, Pi ST (TVI (Cb = s, с) 1hies ,. dc.uo л я .в пФ) у iirtses ТДА,  
wrhoeim са ПК (PO1) олько, так что ttrthuhece trhi omenuu rylitisieptlildcie сор Psfi т та й = РЗЭ 0cno efmoxrtp tuse ТЕС = ир DT oe1nc ут, foro.s .rif vi.tne = Д.Л., уг Т а.е. "ПЛА-icFnoh gte1e, d tuhlae, т T, S s- ifno 1cr.ew {hAitc} привет STH нео-Jno binsc wreivat ш-iinng 1е iaannc TDH,  
семья упорядочиваются в SWPT порядке. Пусть S =  
партия, которая завершается в момент времени C (B,). Кроме того, пусть К = 0, Т = С (В,) для V =  
знак равно  
-. 1 / WPT (В,) для V = 1,, v = т + 1, • • • 'TV,  
е  
{  
μ  
/  
T  
машина времени является более дорогостоящим для более ранних периодов времени, чем для более поздних. Такая ценовая структура  
согласуется с наблюдением, что с большим количеством рабочих мест соревнуются машинного времени в начале  
график, такие периоды времени должны быть назначены более высокую цену.  
??? W1E, .n..o, ВФ, р rtohvenid Леб а (/t.h/)e = или еЕ, tBifceFasl) - ТДВ uwis fthCiefi (rceBa Bft) i1o. Шек Р.Б. tahtcish ФАО с fhoorimcee сделать Ф.Ф. rmomul Taip sliinergsl Е.Б. yjo SBH oowf fianmg itlhya ТФ tfhoer  
нижняя граница точна, когда каждая семья содержит одно задание.  
Теорема 3. Если S = ​​(B1  
е ???  
Доказательство. Когда каждая семья содержит одно задание, эвристический метод следующего раздела генерирует  
график S, в котором партии упорядочены в не порядке убывания значений WPT. Таким образом, мы предполагаем,  
без ограничения общности, что S = (B1, ..., BF), где WPT (B1) <<WPT (Вр).  
Пусть С р = утку + E't ,, t\_s, п + я поинты, обозначает стоимость в лагранжевой задаче планирования  
Jo (бб (F1.) f-) 1to, wcoem obptlaeitne farto tmim (E9) т, т ??? hfoaCrt ftfth ??? е г = РЭБ ILSF ??? sFom (11 e4 аф-5na1md-P it1l у = F + FS1 ', F- wPft-ч I-е • 1r) • е • ??? е '. E, T {. Fw, .. э., FSH} ой, е FO IRR свт ХТ (h1ica0 <ХТ)  
Cft <Cf, т + я при ^ = С (В е), ..., T - 1. Очевидно,  
ПЗУ определение WP T (Bf) я-п (1) ??? = Ш W0e ofbolftra он (п 'Р = й iCaft (tSBh FEF) Ir) Wi, GPHT, Tt - (h1 а- 3n Jd) 1 .s .i Dael OSOF, (1fo1r) ТИС' е = Q UCA ( л Bto.1) г • е (• 1r • o1,)  
Cf, т + 1  
Из формулы (9), заметим, что  
С  
- = 1 / WPT (Вр): индексация партий с использованием значений WPT подразумевает, что ITT, -  
1 / WPT (Bf). Теперь мы установили, что синица, - f.te.4. 1 <1 / WPT (Bf) при Т '= С (В F), ..., Т - 1.  
Подставляя в (10). мы получаем  
Cf, т + 1.- Cft  
 iete п + o1w = s h1o / WW PthTa (т Bcf1t,)>> с 1f, / Tw + я PfoTr (Bt = е) .s ФТФ он, re.fo.re, •, C (B Cf) (B - 1F). , А ... п, Tal о-u1.sly идти с фо} U, R F porre wvihoicuhs  
F  
Урожайность акие CFT> ср, т + 1 при Т = S F ??? е ??? , С (В F) - 1, что и требовалось.  
который устанавливает требуемое неравенство С П <ср, т + I при ^  
W  
анализ, при т '= С.Ф. Pl.? • • •, ??? C (CBF ТФ) 5 - 1 Вт, л thfe ге (Pis is.ofm SEF) / faWmPilTy (Bf'f,), ш - ч - е r0e, е 'E  
-  
Cf, т + 1  
вес  
Приведенный выше анализ показывает, что существует оптимальное решение задачи лагранжевом  
icno nwsthriacinht С.Е. (3в) ч СРК astactihs fBie1d IWS isthc heeqduuallietdy Итно AC osomluptlieotne oaft tthime Ле aCgr (Бан фг) е. ANT phruobs, л eemac. ХТ hoifs tehseta rbelliashxeesd  
что НМТ = Efe Wi ФХ (B F). Q  
Герметичность нижней границы, что мы получаем, когда корректировка множитель используется для поиска  
р, зависит от качества графика С. Первоначально S получается путем применения эвристики  
метод, который представлен в следующем разделе. Тем не менее, всякий раз, когда лучшее решение находится в  
ветвей и границ, S обновляется и мультипликаторы вычисляются из этого нового графика.  
Таким образом, мультипликаторы изменить только тогда, когда улучшенная верхняя граница находится.  
Второй подход для определения значений мультипликаторов использует субградиентного оптимизацию.  
Пусть / (1) быть вектором множителей на итерации 1 этой процедуры, и пусть е) обозначим число  
семей, для которых подстроено или некоторая обработка задания запланировано на интервале [т - 1, т] в  
вычисление LB (А (1)). Первоначально положим р (1) быть мультипликаторы, полученные с помощью мультипликатора  
метод корректировки, wlhite = Re M S IAs xob {Tapini (ЭИ) d ??? + По формуле (f1i) г (SUT FBO г-м LinBg (FAL с (1в)) г) л (ЭИ) б -atc 1h) от каждой семьи, а затем  
секвенирования этих партий в порядке неубывания их значений WPT. После этого мы устанавливаем  
(1 + 1) ??? 1 '  
  
ET-I (е) - 1) 2  
8where UB это верхняя граница минимального значения суммарного взвешенного времени завершения и  
А (I) представляет собой длину скалярной шаг, который удовлетворяет 0 <А (1) <2. В корневой узел ветвей и границ  
дерево поиска. положим А (1) = 2, и выполнить 20 итераций оптимизации Субградиентный, сократить вдвое шаг  
Длина при 5 последовательных итераций не в состоянии улучшить нижнюю границу. Для других узлов поиска  
дерево, мы устанавливаем, а (1), является вектором множителей на родительском узле, мы устанавливаем (1) = 0.1, и выполнять  
3 субградиентные итераций оптимизации, не вдвое сократить длину шага, когда никакого улучшения в нижней части  
оценка наблюдается в течение 2 последовательных итераций.  
4 Эвристический метод  
В этом разделе мы предлагаем эвристический метод, который применяется в корневом узле ветви  
и связанное дерево поиска, чтобы найти верхнюю границу УБ по минимальному значению общее взвешенное  
Время окончания. Наша эвристика использует конструктивную процедуру, основанную на наблюдениях  
Гупта [6], чтобы найти первоначальный график. Descent процедуры затем применяются с целью сокращения  
общее взвешенное время завершения. Метод спуска пытается улучшить текущего раствора  
поиск, в каком-то определенном соответствующим образом окрестности, новое решение, которое имеет более низкую цель  
значение функции. Если такое улучшенное решение найдено, оно становится текущим раствор, из которого  
дальнейшее совершенствование ищется. Если улучшения не возможен, то метод завершается и  
Решение представляет собой локальный оптимум.  
Эвристический метод  
(3 е и м-ф = 1  
Шаг 1. Индекс рабочие места в пределах каждой семьи в SWPT порядке. Набор =  
для F = 1 ...., F. Множество К = 1 и г = 0 (где к первой незаполненной позиции в последовательности  
и г это семейство, к которому принадлежит последнее плановое задание).  
Шаг 2. Выберите семейство Р такой, что аф настолько мал, насколько это возможно. График работы (М.Ф., е) в положении к,  
Множество М.Ф. = МЖ + 1 и к = к + 1. Если м-ф> Н.Ф., установить аф = со; В противном случае установите СИФ =  
Если г = 0, установить G = F. Если G F, установите Ag = (sgwm9s и G = F.  
Шаг 3. Если к <N, а затем перейдите к шагу 2.  
Шаг 4. Нанесите Shift-пакет: алгоритм спуска, который использует окрестность на основе между-  
изменение соседних пар партий.  
Шаг 5. Применить Shift-задание: алгоритм спуска, в котором элементы окрестности хедов  
построненной следующим образом. Для каждой партии, ее последняя работа смещается в обратном направлении до начала следующего  
партия, содержащая работу одного и того же семейства (если это не последняя партия семьи), или его первый  
работа смещается вперед до конца предыдущей партии, содержащей работу одного и того же семейства (если  
это первая партия семьи). Последняя работа последней партии семьи также сдвигается  
в обратном направлении, создавая тем самым новую партию.  
Шаг 6. Применить Shift-пакет.  
График генерируется с помощью эвристического метода согласуется с теоремой 1, как это имеет  
свойство, что рабочие места в пределах каждой семьи упорядочиваются в SWPT порядке.  
95 ветвей и границ алгоритмы  
В этом разделе мы опишем три ветви и связанные алгоритмы. Первый основан на вперед  
ветвление с объективным расщеплению (ООВ), и, по существу, алгоритм Мейсона и Андерсоном  
сын [10]. В корневом узле ветвей и границ дерева поиска, мы применяем процедуру предварительной обработки  
что описано в разделе 2 для формирования композитных рабочих мест и тем самым уменьшить размер проблемы. Также,  
эвристический метод Раздела 4 применяется для генерации начального верхней границы. На каждом узле  
ветвей и границ дерево поиска, правила с 1 по 5 раздела 2 используются в качестве устройств для обрезки деревьев (Правило 3 не  
используется в оригинальном алгоритме Мейсона и Anderson), вместе с динамическим программированием  
Правило доминирования. Глубину первая стратегия поиска используется, где непосредственные потомки узла  
рассматриваются в порядке неубывания их нижних границ.  
Вторая ветвь и связанный алгоритм использует вперед ветвления и регулировки множитель  
(ФМБА). Помимо способа вычисления нижних границ, он идентичен первому алгоритму.  
Первоначально нижняя граница LB (р) использует мультипликаторы, которые получаются из эвристического графика.  
В расчете нижняя граница, верхняя граница также вычисляется: формы из партии  
рабочие места и той же семьи, которые планируется смежно в нижней вычисления ограничивающего,  
использовать теорему 2 к последовательности этих партий, и оценить полученный график. Если в результате  
верхняя граница меньше, чем текущая верхняя граница, то UB обновляется, и мультипликаторы основаны  
на этом графике улучшения в последующих более низких вычислений ограничивающими.  
Наша третья ветвь и связанный алгоритм использует двоичную правило ветвления и субградиент оптими-  
ции (BBSO). В корневом узле, предварительная обработка процедура раздела 2 применяется. На каждом узле  
дерева поиска, мы вычислим верхнюю границу путем построения двух эвристических графики. Первый  
Эвристический формирует партию из каждого семейства, а затем последовательности этих партий в порядке неубывания  
их значений WPT. Во втором эвристики, задания упорядочены в SWPT порядке. лагранжиан  
нижняя граница вычисляется на каждом узле, как описано в разделе 3, где оптимизация субградиент является  
используется для поиска значений мультипликаторов. Наше правило ветвления выбирает пару рабочих мест (J - 1, е) и (к, е),  
где (у, е) Е Н и J> 1, и ограничивает эти рабочие места либо содержаться в том же пакете,  
или которые должны содержаться в разных пакетах, в отличие от переднего разветвлением, это двоичное ветвление позволяет  
возможность принятия важных решений на более высоких уровнях ветвей и границ дерева поиска.  
Для отрасли, в которой рабочие места (к - 1, е) и (к, е) должны содержаться в том же пакете, мы формируем  
композит работу из (J - 1, е) и (к, е). В другой ветви, в которой работа (у, е) начинается новый  
партия, мы разбиваем рабочие места семейства Р на два семейства, содержащих рабочие места (1, е), ..., (J - 1, е) и  
рабочие места (J, F), ..., (N F, F) соответственно. Мы выбираем работу (J, F), так что лагранжиан проблема запла-  
вакансии üles (J - 1, J) и (к, е), чтобы закончить в момент времени Т и Т, соответственно, где т 'p3 е <т и  
г (J, T 'р3 г (J, T) как можно больше (Z (J, T' + р е) - Z (J, т) получается из динамического про-  
граммирования рекурсию для лагранжевой задачи и обеспечивает меру стоимости не расщеплению  
Семейство F так, что работа (у, е) начинает новую партию). Для этого бинарного правила ветвистый, начальные эксперименты  
с нашими лагранжевых нижними границами, где мультипликаторы получаются с помощью оптимизации субградиентного  
и с помощью метода изменения множителя, показывают, что оптимизация подход субградиент  
превосходит. Как и в двух других алгоритмов, используется в глубину стратегия поиска.  
106 Вычислительная опыт  
В этом разделе мы сообщаем о результатах вычислительных тестах для оценки эффективности  
три ветви и связанные алгоритмы. Алгоритмы FBOs и ФМБА были закодированы в ANSI-C, в то время как  
BBSO была закодирована в FORTRAN 77. Все алгоритмы были работать на компьютере HP 9000/715. Для  
FBOs. вычисление прекращается, если предел в 50 000 узлов превышен, в то время как предельный срок  
600 секунд используется для BBSO. Без ограничений применяются для ФМБА (все проблемы могут быть решены с помощью  
разумные вычислительные ресурсы).  
тестовые задачи с 30, 40 и 50 рабочих мест, а также с 4, 6, 8 и 10 семей были получены в качестве  
следующим образом. Для каждой комбинации N и F, рабочие места равномерно распределены по семьям, так  
что каждая семья содержит либо [N / FJ или [рабочих мест N / Fl. Время обработки и вес случайным образом  
порожденные целыми числами от равномерного распределения, определенные на [1,10]. Поскольку размер времени наладки  
по отношению к времени обработки может повлиять на проблему 'твердость', мы получили проблемы с малым (S),  
средний (M) и большой (L) раз настройки. время наладки среды генерируется случайным образом целыми числами  
от равномерного распределения определенной на [1,10]. Сформировав экземпляр со средней настройки  
раз, s! (F = 1,, F), соответствующие экземпляры с маленькими время переналадки RS F / 21 и с большим  
2SF были построены раз настройки. Для каждой из 12 комбинаций N и F, 50 тестовых задач  
были созданы с малыми, средними и с большими времени наладки.  
Наши проблемы тестирования охватывают целый ряд различных сценариев. Число рабочих мест в семейных диапазонах  
от 3 (для N = 30 и F = 10) до 12 и 13 (при N = 50 и F = 4). Кроме того, различные настройки  
диапазон времени считается. Проблемы, связанные с очень малыми временами наладки, вероятно, оказывают незначительное влияние,  
так что оптимальная последовательность заданий будет близка к SWPT. С другой стороны, партии будет  
формируется из полных семей, когда время наладки очень велики. Оба этих крайних случаев  
вероятно, будет относительно легко решить. Возможные расписания в наших случаях может включать машину  
проходит время установки лишь около 4% времени (для N = 50 и F = 4 при выборе набора параметров раза  
маленькие и есть только один установка для каждой семьи), или около 67% времени (для больших  
раз настройки, когда время установка понесены для каждого задания).  
В таблице 1 приведены результаты расчетов для трех ветвей и связанных алгоритмов. Для каждого  
комбинация N и F. раз в среднем вычисления в секундах (ACT), среднее число узлов  
(ИНС) и количество нерешенных проблем (NU) приведены для случаев с малыми, средними  
и большие интервалы времени наладки. Когда есть нерешенные проблемы, значения перечислены в ACT и ANN  
являются нижние оценки истинных средних значений.  
Заметим прежде всего, из таблицы 1 следует, что существует значительное число нерешенных проблем  
Алгоритмы FBOs и BBSO. Тем не менее, ФМБА решает все проблемы тестирования, не создавая большой  
деревья поиска или требует слишком много времени вычислений. Например, среди 600 тестовых задач  
с установленным планом времен средних, только 6 требуют более 3 минут времени вычислений.  
Вычислительные затраты существенно зависит от F алгоритмов FBOs и BBSO, но  
достаточно стабильной для ФМБА. Алгоритм FBOs захватывает часть структуры проблемы через свои  
правила доминирования. что позволяет ему решать проблемы, связанные с небольшим числом семей. Однако,  
цель расщепления связаны слишком слаб, чтобы контролировать взрыв размера дерева для больших проблем,  
особенно, когда есть много семей. Алгоритм BBSO также захватывает часть проблемы