Available online at www.sciencedirect.com
ScienceDirect
Computers and Mathematics with Applications 56 (2008) 151–159
An International Journal
computers & mathematics with applications
www.elsevier.com/locate/camwa

Improved hybrid simulated annealing algorithm for navigation scheduling for the two dams of the Three Gorges Project

Xiaopan Zhang[a], Xiaohui Yuan[b,*], Yanbin Yuan[c]

[a] Institute of System Engineering, Huazhong University of Science and Technology, Wuhan, 430074, China
[b] School of Hydropower & Information Engineering, Huazhong University of Science and Technology, Wuhan, 430074, China
[c] School of Resources & Environmental Engineering, Wuhan University of Technology, Wuhan, 430070, China

Received 30 June 2006; received in revised form 16 October 2007; accepted 17 November 2007

Abstract

The purpose of this paper is to deal with the optimal navigation co-scheduling (NCS) for the two dams of the Three Gorges Project in China, i.e., the Three Gorges Dam and the Gezhouba Dam. The co-scheduling includes the operational scheduling for all five locks of the two dams, and the dispatching scheduling to ships applying to pass the dams. Compared with the flexible manufacturing system (FMS), a mixed-integer nonlinear programming model is designed for the NCS, and an improved hybrid algorithm of simulated annealing and local search is addressed to obtain optimal scheduling. Experiments based on real historical execution data show that the approach is feasible.

Keywords: Simulated annealing; Local search; Mixed-integer nonlinear programming; Optimal navigation co-scheduling; Three gorges project

1. Introduction

The famous Three Gorges Project includes the Three Gorges Dam upriver and Gezhouba Dam downriver. The Gezhouba Dam has three single-step locks, and the Three Gorges Dam has two five-step locks, which are shown in Fig. 1.

The Navigation Co-Scheduling (NCS) involves lock-operation scheduling that decides the operation timetable for each lock, and ship-dispatch scheduling that decides the passage timetable for each ship. Practically there is a requirement for the navigation department to obtain an optimal plan for NCS for a period under some objectives. It is very important to promote the navigation capability of the golden waterway. In order to represent the details of NCS there are some concepts that should be explained. Lock service is defined as the following process: the lock gate opens, then ships enter the lock chamber and are transferred through the dam. There are four important considerations in the process. The first is when the process starts, i.e., the lock gate opens, which is called the service point. The second is the direction of ships that are transferred by the lock service. Obviously all ships that are transferred by the same lock service have the same navigation direction. This property is called the service direction. The third is the

* Corresponding author. Tel.: +86 27 87556527; fax: +86 27 67884228.
E-mail address: yxh71@163.com (X. Yuan).

doi:10.1016/j.camwa.2007.11.041

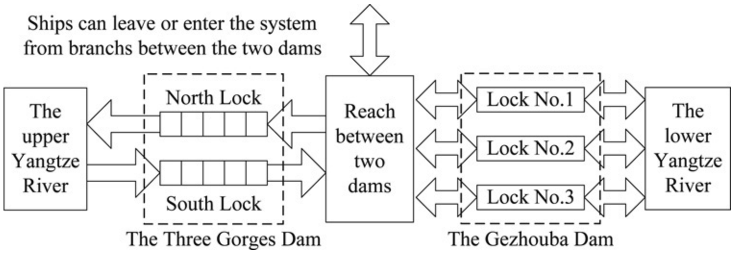

Fig. 1. The architecture of the navigation system in the Three Gorges Project.

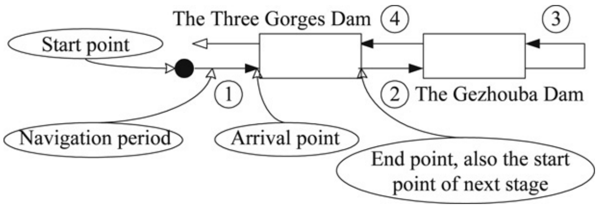

Fig. 2. A ship with four stages.

Table 1
The similarities between NCS and FMS

Conception in NCS	Similar conception in FMS
Lock	Machine
Ship	Job
Stage	Operation
Service interval of lock services	Processing time
Conversion time of lock	Setup time
Waiting time of ship	Tardiness

time taken by a lock to transfer a chamber of ships through dam. In NCS this is called the transfer time. The last point is the minimal time interval, named the service interval, between the service points of two consecutive lock services. This is shorter than the transfer time in the five-step locks, because they running as if through a pipe, and is longer than transfer time in the single-step locks, because of they run like an elevator. If two consecutive lock services have the same direction in the five-step locks or different directions in the single-step locks there is an additional conversion process between them, called lock conversion.

The navigation route of a ship can be divided into stages corresponding to the dams on it. Fig. 2 shows an example of a ship with four stages. The concepts in elliptical regions denote the time of key points in stages

There are three objectives for the optimal NCS plan, i.e.,

(a) Minimize the total weighted tardiness of ships. The tardiness of a ship is an increasing function of its waiting time. The objective is equal to the sum of the tardiness of each ship weighted by its degree of importance and occupied area.
(b) Minimize the quantity of lock conversion. Empty lock operation in lock conversion not only wastes water but also harms lock equipment.
(c) Maximize the workload balance of the three locks of the Gezhouba Dam within scheduling period. The workload is measured by the sum of all lock services. Each lock of the Gezhouba Dam has a predefined optimal workload rate within the total of the three locks. A more balance workload of a lock means its actual rate is closer to the optimal value.

There are many similarities between NCS and a Flexible Manufacturing System (FMS) as shown in Table 1.

But the NCS is more complex than the FMS because of the lock transferring a batch of ships simultaneously, and the arrangement of ships in the lock chamber is also a NP-hard bin-packing problem. There have been some related studies about the problem, such as the arranging problem in a lock chamber [1–3], multi-attribute decisions on ship dispatching [4] and the scheduling problem of the Gezhouba Dam [5], but studies about co-scheduling for the five locks are lacking. Because of the similarities between FMS and NCS, the available knowledge about FMS is helpful. A global mixed-integer linear programming model of FMS has been presented in [6]. Since FMS has NP-complete complexity, heuristic algorithms such as simulated annealing [7,8] and genetic algorithms [9,10] are very suitable for large-scale problems.

2. Mathematical model

2.1. Symbols and parameters

I: The index set of ships.
i: The index of ship $i \in I$.
j: The index of the stage.
$j(i)$: Total stages of the ship i.
$U = \{(i, j) \mid i \in I, 0 \le j < j(i)\}$: (i, j) means scheduling units, which denotes ship i at stage j.
v_{ij}: The navigation direction of ship $i$at stage j, 0 for upriver and 1 for downriver.
τ_{ij}: The length of navigation period when ship i is at stage j.
l_{ij}, w_{ij}: The length and width of ship i at stage j.
α_{ij}: The degree of importance of ship i at stage j.
k: The index of the lock.
$K = \{0, 1, 2, 3, 4\}$: The index set of locks, where 0, 1, 2, 3, 4 denote the South Lock and North Lock of the There Gorges Dam, and Lock No. 1, No. 2 and No. 3 of the Gezhouba Dam, respectively.
$K(i, j)$: The set of available locks for ship i at stage j for passing, $K(i, j) \in K$.
m: The sequence number of lock service of a lock.
$m(k)$: The maximum lock services of lock k.
(k, m): The mth lock service of lock k.
$G = \{(k, m) \mid k \in K, 0 \le m < m(k)\}$: The set of lock services.
l_k, w_k: The available length and width of lock k.
p_k: The transfer time of lock k.
t_k^s: The service interval of lock k.
t_k^c: The conversion time of lock k.
β_k: The optimal workload balance rate of lock k. $\beta_0 = \beta_1 = 0$, $\beta_2 + \beta_3 + \beta_4 = 1$.
t_b, t_e: The beginning and end of the scheduling period.
$\mu(x)$: The step function, i.e., if $x > 0$, $\mu(x) = 1$, otherwise $\mu(x) = 0$.

2.2. Variables and tool functions of scheduling

$r_{km} \in \{0, 1\}$: 1, if lock service (k, m) is operational; 0, otherwise.
$t_{km} \in R^+$: The service point of lock service (k, m).
$d_{km} \in \{0, 1\}$: The service direction of lock (k, m), 0 for upriver and 1 for downriver.
$z_{ijkm} \in \{0, 1\}$: 1, if ship i is transferred by lock service (k, m) at its stage j.
$x_{ij} \in \mathbf{R}^+$: The x-coordinate of ship i placed in the lock chamber at its stage j.
$y_{ij} \in \mathbf{R}^+$: The y-coordinate of ship i placed in the lock chamber at its stage j.

2.3. Constraints and cost functions

Lock services sequencing constraints. These five constraints ensure that all lock services in the same lock are listed by ascending order of time and with proper intervals:

$$r_{km-1} \ge r_{km}, \quad 1 \le m < m(k) \tag{1}$$

$$t_{km-1} z_{ijkm} \le t_{km} z_{ijkm}, \quad 1 \le m < m(k) \tag{2}$$

$$t_b z_{ijkm} \le t_{km} z_{ijkm} < t_e z_{ijkm}, \quad 0 \le m < m(k) \tag{3}$$

$$r_{km}(t_k^s + t_k^c |d_{km} - d_{km-1}|) \le r_{km}(t_{km} - t_{km-1}), \quad 0 \le m < m(k), k \in \{0, 1\} \tag{4}$$

$$r_{km}[t_k^s + t_k^c (1 - |d_{km} - d_{km-1}|)] \le r_{km}(t_{km} - t_{km-1}), \quad 0 \le m < m(k), k \in \{2, 3, 4\}. \tag{5}$$

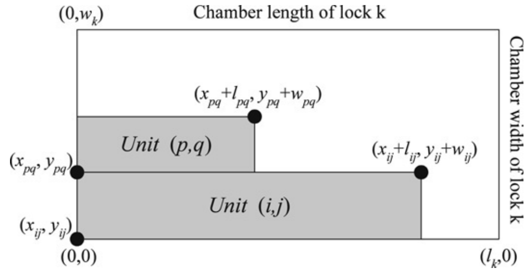

Fig. 3. Ship arrangement in a lock chamber. Two ships for example, i.e., unit (i, j) and (p, q).

Navigation constraints. These six constraints ensure that the result of ship dispatch scheduling is reasonable:

$$z_{ijkm} r_{km} = z_{ijkm} \tag{6}$$

$$\sum_{k \in K(i,j)} \sum_{m=0}^{m(k)-1} z_{ijkm} \le 1 \tag{7}$$

$$\sum_{k \in (K-K(i,j))} \sum_{m=0}^{m(k)-1} z_{ijkm} = 0 \tag{8}$$

$$\sum_{(k,m)\in G} z_{ijkm} \le \sum_{(k,m)\in G} z_{i(j-1)km}, \quad j \ge 1 \tag{9}$$

$$d_{km} z_{ijkm} = v_{ij} z_{ijkm} \tag{10}$$

$$\sum_{(k,m)\in G} z_{ijkm} \times \sum_{(k,m)\in G} z_{i(j-1)km}(t_{km} + p_k + \tau_{ij}) \le \sum_{(k,m)\in G} z_{ijkm} t_{km}. \tag{11}$$

Packing constraints. These three constraints ensure that none of the ships in the same lock chamber overlap, as shown in Fig. 3.

$$0 \le z_{ijkm} x_{ij} \le z_{ijkm}(l_k - l_{ij}) \tag{12}$$

$$0 \le z_{ijkm} y_{ij} \le z_{ijkm}(w_k - w_{ij}) \tag{13}$$

$$z_{ijkm} z_{pqkm} \mu(x_{pq} + l_{pq} - x_{ij}) \mu(y_{pq} + w_{pq} - y_{ij}) \mu(x_{ij} + l_{ij} - x_{pq}) \mu(y_{ij} + w_{ij} - y_{pq}) = 0. \tag{14}$$

Obj. 1. *Tardiness cost of ship*

$$Z(i, j) = l_{ij} w_{ij} \alpha_{ij} (\Delta t_{ij} + \Delta t_{ij}^2). \tag{15}$$

The waiting time Δt_{ij} should be calculated as follows:

$$\Delta t_{ij} = \begin{cases} \left[\sum_{(k,m)\in G} z_{ijkm} t_{km} - \sum_{(k,m)\in G} z_{i(j-1)km}(t_{km} + p_k + \tau_{ij})\right] \Big/ (t_e - t_b) \\ \qquad \text{if } \sum_{(k,m)\in G} z_{ijkm} \times \sum_{(k,m)\in G} z_{i(j-1)km} = 1, \\ \left[t_e - \sum_{(k,m)\in G} z_{i(j-1)km}(t_{km} + p_k + \tau_{ij})\right] \Big/ (t_e - t_b) \\ \qquad \text{if } \sum_{(k,m)\in G} z_{ijkm} + \sum_{(k,m)\in G} z_{i(j-1)km} = 1, \\ 0, \quad \text{else.} \end{cases} \tag{16}$$

Obj. 2. *Conversion cost of lock service*

$$C(k, m) = \begin{cases} 3 r_{km} l_k w_k |d_{km} - d_{km-1}| & k \in \{0, 1\} \\ r_{km} l_k w_k (1 - |d_{km} - d_{km-1}|) & k \in \{2, 3, 4\}. \end{cases} \tag{17}$$

Obj. 3. *Unbalanced workload cost of lock*

$$E(k) = \begin{cases} \left| \dfrac{\sum_{m=0}^{m(k)-1} r_{km}}{\sum_{a=3}^{5} \sum_{m=0}^{m(a)-1} r_{am}} - \beta_k \right| \times \left(\sum_{m=0}^{m(k)-1} r_{km} l_k w_k \right) & k \in \{2, 3, 4\} \\ 0 & k \in \{0, 1\}. \end{cases} \tag{18}$$

NCS is defined as a single objective program with weighted sum of the three costs:

$$\min J(U, K) = \alpha_1 \sum_{(i,j)\in U} Z(i, j) + \alpha_2 \sum_{k\in K} \left[c_k \sum_{m=0}^{m(k)-1} C(k, m) \right] + \alpha_3 \sum_{k\in K} e_k E(k). \tag{19}$$

3. Hybrid algorithm

In this section, we propose a hybrid algorithm of simulated annealing and local search. The solution space is separated into local domains by r_{km} and d_{km}, i.e., all the solutions in the same domain have the same values of r_{km} and d_{km}. In each domain simulated annealing is used to search the local optimal values of t_{km}. The ship-dispatch scheduling is obtained by a process represented in 3.1 according to r_{km}, t_{km} and d_{km}.

U: ship-dispatch scheduling, including z_{ijkm}, x_{ij} and y_{ij},
K: lock-operation scheduling, including r_{km}, t_{km} and d_{km},
U^*: the local best ship-dispatch scheduling,
K^*: the local best lock-operation scheduling,
J^*: the local best cost,
U_{opt}: the global best ship-dispatch scheduling,
K_{opt}: the global best lock-operation scheduling,
J_{opt}: the global best cost,
LOOP: maximum loop counts.

(a) Make an initial solution of r_{km}, t_{km} and d_{km}, and obtain the solution of z_{ijkm}, x_{ij} and y_{ij} by the ship-dispatching process. Let $J_{\text{opt}} = J(U, K)$, and $(U_{\text{opt}}, K_{\text{opt}}) = (U, K)$. Initialize $loop = 0$.
(b) Search the local best solution (U^*, K^*) by the use of simulated annealing. If $J^* < J_{\text{opt}}$, $J_{\text{opt}} = J^*$, $(U_{\text{opt}}, K_{\text{opt}}) = (U^*, K^*)$.
(c) $loop = loop + 1$. If $(loop \geq LOOP)$, stop and return $(U_{\text{opt}}, K_{\text{opt}})$.
(d) Make the heuristic adjustment to r_{km} and d_{km}, initialize t_{km}, and then go to (b).

The algorithm in the local domain imitates the improved simulated annealing in [7], i.e.
SUM: total number of lock services to be processed,
ULITA: upper limit of number of moves at a particular temperature,
Total: a counter, total number of moves at a particular temperature,
acts: a counter, number of accepted moves at a particular temperature,
r: temperature reduction factor in the annealing process,
q: predetermined fraction of accepted moves at a certain temperature,
t: number of times that the temperature is decreased in the annealing process,
f_c: a counter that is used to check whether the solution is 'frozen' or not.

(a) *Initialization*. Obtain an initial solution K for lock-operation scheduling. Initialize parameters r, q, t. Set the initial temperature T. Set counter $f_c = 0$, $Total = 0$, $acts = 0$, tabu list $\Omega = \Phi$. Calculate the final temperature $T_f = r^t \times T$.
(b) *Ship dispatching*. Obtain a solution U of ship-dispatch scheduling by simulating the actual navigation process and dispatch ships for lock-operation scheduling K. Set best co-scheduling $(U^*, K^*) = (U, K)$.

(c) *Neighbor search*. Select a neighbor solution K^c of K. Obtain the ship-dispatch solution U^c by doing *Ship dispatching* for K^c.
(d) Compute $\Delta = J(U^c, K^c) - J(U, K)$.
(e) If $\Delta \leq 0$, set $(U, K) = (U^c, K^c)$ and $acts = acts + 1$. Compute $\Delta_b = J(U, K) - J(U^*, K^*)$. If $\Delta_b < 0$, set $(U^*, K^*) = (U, K)$, and $f_c = 0$.
(f) If $\Delta > 0$, select a random variable $X \sim U(0, 1)$. If $e^{-\Delta/T} > X$, set $(U, K) = (U^c, K^c)$ and $acts = acts + 1$.
(g) $Total = Total + 1$. If ($Total < ULITA$) and ($acts < SUM$) return to (c).
(h) Compute $pc = acts/Total$, if $pc < q$, set $f_c = f_c + 1$.
(i) Set $T = r \times T$, $Total = 0$, $acts = 0$.
(j) If ($T \leq T_f$) or ($f_c \geq 5$), stop and return (U^*, K^*). Else return to (c).

3.1. Ship dispatching

N_k: number of operational lock services in lock k, $N_k = \sum_{m=0}^{m(k)-1} r_{km}$,
$c(k)$: order number of the current lock service of lock k.
$\tau_{ij}^a \in R^+$: available time when the scheduling unit (i, j) can be dispatched,
$\delta_{ij} \in \{0, 1, 2\}$: status of stage j of ship i, $\delta_{ij} = 0$, if stage j hasn't started; $\delta_{ij} = 1$, if ship i is at stage j currently; and $\delta_{ij} = 2$, if stage j has ended.
U_{km}: set of available ships for (k, m). $U_{km} = \{(i, j) \mid k \in K(i, j), \delta_{ij} = 1, \tau_{ij}^a \leq t_{km}, v_{ij} = d_{km}\}$.

The set U_{km} ensures that only those ships can be dispatched that satisfy the navigation constraints until the lock service (k, m) starts. We suppose that the difference between the arrival point of unit (i, j) and the service point of lock service (k, m) equals t_{ij}, and the ship that has not been selected, i.e., $z_{ijkm} = 0$, will be transferred after the time of $\Delta\tau_{ij}$. Then the tardiness costs of ships in U_{km} are as follows:

$$\sum_{(i,j)\in U_{km}} [z_{ijkm}\lambda_{ij}(t_{ij} + t_{ij}^2) + (1 - z_{ijkm})\lambda_{ij}[(t_{ij} + \Delta\tau_{ij}) + (t_{ij} + \Delta\tau_{ij})^2]] \tag{20}$$

where in $\lambda_{ij} = l_{ij} w_{ij} \alpha_{ij}$.

Linearizing the above formula, we can get

$$f(U_{km}) = \sum_{(i,j)\in U_{km}} [\lambda_{ij}(t_{ij} + t_{ij}^2) + \lambda_{ij}\Delta\tau_{ij}(1 - z_{ijkm})(1 + 2t_{ij})]. \tag{21}$$

According to Obj. 1, we should minimize $f(U_{km})$, so the dispatching rule should be the following integer linear program:

$$\max \xi(U_{km}) = \sum_{(i,j)\in U_{km}} z_{ijkm}\lambda_{ij}\Delta\tau_{ij}(1 + 2t_{ij}) \quad \text{s.t. (13), (14), (15).} \tag{22}$$

This is a 2-D bin-packing problem that has studied in [1–3].

Based on this rule, the ship dispatching process can be represented as the follows:

(a) For each scheduling unit (i, j), set the initial τ_{ij}^a as the time at which it wants to pass the dam in its.
(b) For each scheduling unit (i, j), if $j = 0$, set initial $\delta_{ij} = 1$, otherwise set initial $\delta_{ij} = 0$. Set $c(k) = 0$.
(c) If $c_k \geq N_k$ for every lock, or $\delta_{ij} = 2$ for every scheduling unit, stop and output the solution of ship dispatching.
(d) Select the lock service $(k, c(k))$ within all locks where $c(k) < N_k$ and $t_{kc(k)}$, i.e., th service point, is the smallest. Then dispatch ships by the rule (22).
(e) For each unit (i, j) that has dispatched to $(k, c(k))$, i.e., $(i, j) \in U_{kc(k)}$ and $z_{ijkc(k)} = 1$, set $\delta_{ij} = 2$, $\delta_{ij+1} = 1$, $\tau_{ij+1}^a = \max\{\tau_{ij+1}^a, (t_{kc(k)} + p_k + \tau_{ij+1})\}$. Set $c(k) = c(k) + 1$, go to (c).

Table 2
Main parameters and initial values

Parameter	Value
Scheduling period	24 h
c_k	1.0 for $k = 2, 4$; 20.0 for $k = 0, 1$; 8.0 for $k = 3$
e_k	1.0 for $k = 2, 3, 4$
$\alpha_1, \alpha_2, \alpha_3$	50.0, 2.0, 1.0.
$\beta_2, \beta_3, \beta_4$	0.206, 0.256, 0.538
$m(1), m(2), m(3), m(4), m(5)$	13, 17, 16, 20, 42
LOOP	20
Cooling rate r	0.9
Temperature decreasing times	20
SUM	110
ULITA	3 **SUM*
Initial temperature T	2×10^4
Initial r_{km}	1
Initial t_{km}	$t_{k0} = 1, t_{km} = t_{km-1} + t_k^s$
Initial $d_{km}, k = 0, 1$	$d_{00} = 1, d_{10} = 0, d_{km} = d_{km-1}$
Initial $d_{km}, k = 2, 3, 4$	$d_{k0} = 0, d_{km} = \|1 - d_{km-1}\|$

3.2. Heuristic adjustment

There are two kinds of heuristic adjustment acting on lock-operation scheduling. One is for moving the local solution domain by adjusting the values of r_{km} and d_{km}, and the other is for neighbor moving of the simulated annealing solution in a local domain by adjusting the values of t_{km}. The first adjustment is made up of three operations, i.e.,

Remove lock service. Select the operational lock service within all locks that has the maximum utilization cost. Then remove the lock service for its own lock. Calculate the utilization cost U_{km} for each operational lock service in lock k, and the sum of U_{km}, i.e., U_k. Select an operational lock service with the probability of U_{km}/U_k and remove it. Suppose the lock service is (k, m), removing it means that from $n = m$ to $n = N_k - 1$, where N_k denotes the number of operational lock services in lock k, set $r_{kn} = r_{kn+1}$, $t_{kn} = t_{kn+1}$, and $d_{kn} = d_{kn+1}$.

Add lock service. Calculate the time interval between each two consecutive operational lock services, $\Delta(k, m)$, in each lock. That is if $m < N_k - 1$, $\Delta(k, m) = t_{km+1} - t_{km} - 2t_k^s$, and if $m = N_k - 1$, $\Delta(k, m) = t_e - t_{km} - t_k^s$. Let $\Delta = \sum_{k=1}^{5} \sum_{m=0}^{N_k} \Delta(k, m)$, select (k, m) with the probability of $\Delta(k, m)/\Delta$, and add a new operational lock service after it.

Convert lock service. Select an operational lock service for the locks in the Gezhouba Dam randomly, and change its service direction, which means that if $t_{km} - t_{km-1} \geq t_k^s + t_k^c|d_{km} - d_{km-1}|$, set $d_{km} = |1 - d_{km}|$.

And the second adjustment includes two operations, i.e.,

Increase service point. Calculate $\psi_{km} = 1/\xi(U_{km})$ for each operational lock service in lock k, and the sum of ψ_{km}, i.e., ψ_k. Select an operational lock service with the probability of ψ_{km}/ψ_k to increase its service point by some time step.

Decrease service point. Calculate the total waiting costs C_{km} of all ships in each operational lock service in lock k, and the sum of C_{km}, i.e., C_k. Select an operational lock service with probabilities of C_{km}/C_k to decrease its service point by some time step.

4. Experimental results

We have tested the algorithm with real historical execution data. Typical examples for 4 days in a busy navigation season of the water area are presented in Tables 2 and 3.

Remark 1. The initial lock operation scheduling, i.e., r_{km}, t_{km}, d_{km}, means that the locks work with full load and in a regular manner.

Table 3
Count of scheduling units

	Day 1	Day 2	Day 3	Day 4
Applying to pass TGD upward	42	75	96	199
Applying to pass TGD downward	55	61	86	101
Applying to pass GD upward	48	78	96	213
Applying to pass GD downward	86	105	133	141

TGD (he Three Gorges Dam), GD (the Gezhouba Dam).

Table 4
Statistics of results, where *IMP_RATE*= (initial cost − min cost)/initial cost

	Day 1		Day 2		Day 3		Day 4	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Computation (s)	155.9	5.2	447.5	12.9	664.2	20.3	3 243.2	116.4
Min $J(U, K)$	14 795.8	937.6	14 078.2	1502.5	23 775.9	51.3	56 687.9	911.3
IMP_RATE (%)	16.9	5.6	44.2	7.4	1.8	0.2	7.0	1.8

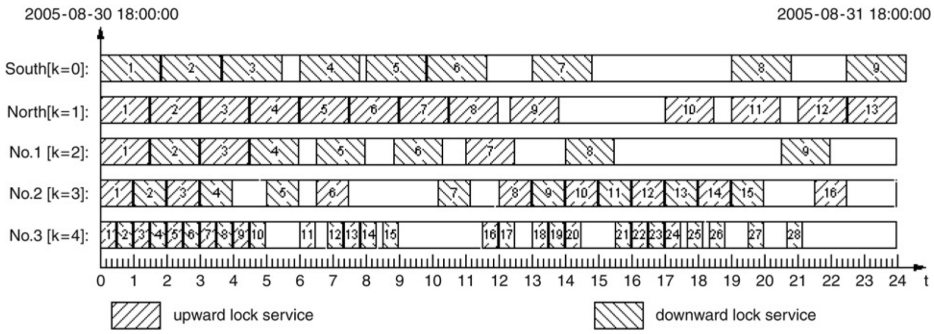

Fig. 4. The best lock operation plan for Day 1.

Table 4 was obtained on a Pentium-IV 2.4 GHz, 512 MB memory and Windows 2000 system. Each test has been run 10 times.

For Days 1 and 2, the algorithm improves the initial solution remarkably. But since there are a lot of scheduling units in Days 3 and 4, the workload of locks should be nearly to full load, i.e., the initial solution, so the improvements are not remarkable in these 2 days. And Fig. 4 shows an example of the lock operation plan.

The hatched blocks denote operational lock services, where the width denotes the service interval and the hatching denotes the service direction.

We randomly select five results of the four experiments to show the updating traces of costs in Fig. 5.

In most of these traces the global minimal cost can be obtained within 15 rounds of local search, and the local minimal cost becomes relatively stable after 15 rounds. So for practice, the parameter, *LOOP*, can be set to 15.

5. Conclusions

A mixed-integer nonlinear programming model with three objectives has been made for the NCS problem of the Three Gorges Project. And a hybrid algorithm of simulated annealing and local search is designed to deal with the problem. Finally some experimental results are presented, which show that the algorithm is feasible. This algorithm has been used in the practical navigation scheduling system, which was put into operation in January 2006.

Nevertheless, the shortcomings of the model and approach are only focused on determinate navigation conditions. Because the condition of the water area in the Three Gorges Project is very complex, it is necessary to expand the model to the stochastic field and develop its learning ability to accelerate computation.

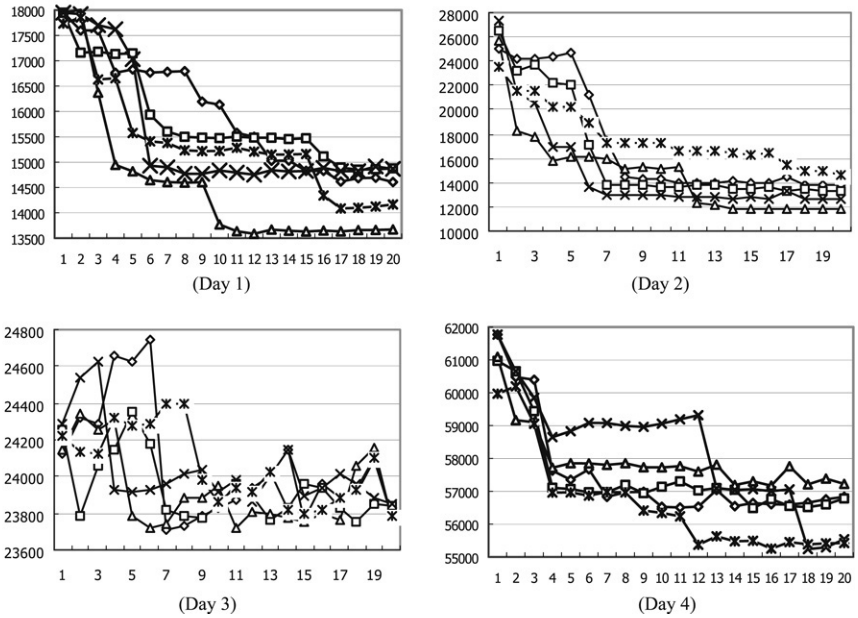

Fig. 5. Each sub-figure shows five updating traces of a day, each mark on traces of which denotes the minimal cost ($J(U, K)$) in a local search process.

Acknowledgements

The authors gratefully acknowledge financial support from the National Natural Science Foundation of China under grants no. 50779020, 40572166, 50309013 and 50409010. The authors would also like to thank Professor Huan Qi from Huazhong University of Science & Technology for his contributions.

References

[1] Y.-F. Liu, H. Qi, Application of DFS algorithm in the arranging of three gorges permanent lock chamber, Computer Engineering 28 (2002) 224–226.
[2] Y.-F. Liu, H. Qi, The two-dimension optimization arranging heuristic algorithm and its application in the Yangtse gorges permanent ship lock decision system, Computer and Modernization 1 (2002) 1–3.
[3] F.-Y. Lu, H. Qi, J. Cao, Research and application of computer in the arranging of three gorges permanent lock chamber, Application Research of Computers 17 (2000) 65–67.
[4] W. Lai, H. Qi, The MADM of three gorges ship gates running, Control and Decision 17 (2002) 163–166.
[5] J.-N. Du, S.-M. Yu, Dynamic programming model and algorithm of shiplock scheduling problem, Computer and Digital Engineering 31 (2003) 47–50.
[6] Jiyin Liu, B.L. MacCarthy, A global MILP model for FMS scheduling, European Journal of Operational Research 100 (1997) 441–453.
[7] Chinyao Low, Yukling Yip, Tai-His Wu, Modelling and heuristics of FMS scheduling with multiple objectives, Computers & Operations Research 33 (2006) 674–694.
[8] Chinyao Low, Tai-His Wu, Mathematical modelling and heuristics approaches to operation scheduling problems in an FMS environment, International Journal of Product Research 39 (2001) 689–708.
[9] Beatrice M. Ombuki, Mario Ventresca, Local search genetic algorithms for the job shop scheduling problem, Applied Intelligence 21 (2006) 99–109.
[10] Ben Hadj-Alouane, James C. Bean, Katta G. Murty, A hybrid genetic/optimization algorithm for a task allocation problem, Journal of Scheduling 2 (1999) 189–201.