УДК 656.614:656.073.235

Многоступенчатый генетический алгоритм для предварительной сортировки контейнеров

ЛИ ХАОЮАНЬ
Li HAOYUAN

САН ДОНГШИ
Sun DONGSHI

Ли Хаоюань *– доктор наук, доцент, заведующий кафедрой бизнес-администрирования, Университет информатики Neusoft, Далянь, Китайская Народная Республика.*
Сан Донгши *– преподаватель, магистр наук, руководитель логистической группы, Университет информатики Neusoft, Далянь, Китайская Народная Республика.*

Multi-Stage Genetic Algorithm for Container Pre-Marshalling Problem

(текст статьи на англ. яз. – English text of the article – p. 12)

Чтобы решить задачу предварительной сортировки на контейнерной площадке, по мнению авторов, требуются математическая модель программирования и многоступенчатый генетический алгоритм с двоичной схемой кодирования. В предлагаемом алгоритме целевая функция образована посредством определения коэффициента нечеткости; несколько эвристических операторов обеспечивают разработку необходимых методических и операционных действий. Как могут быть достигнуты лучшие решения, показано на примере двух исследований, подтверждающих эффективность генетического алгоритма с бинарным кодированием в качестве средства оптимизации при построении модели предварительной сортировки контейнеров.

Ключевые слова: морские перевозки, терминал, контейнер, предварительная сортировка, генетический алгоритм, бинарное кодирование, математическая модель, эвристика.

По общему правилу, обычные экспортные контейнеры доставляются в контейнерный терминал в случайной последовательности в течение семи дней до отправления судна. Если ёмкости загружаются на контейнеровоз в том порядке, в котором они располагались в терминале, это скорее всего приведет к большой проблеме. Когда контейнеровоз прибудет в место назначения, может обнаружиться, что контейнеры, принадлежащие этому судну, оказались под контейнерами, принадлежащими другому. В такой ситуации все контейнеры другого судна нужно перегруппировать и перезагрузить, что приведет к немалым затратам времени.

Чтобы решить подобную проблему, на первый план выдвигается понятие «предварительной сортировки». Предварительная сортировка означает, что контейнеры должны быть перегруппированы заранее. То есть прежде чем они будут загружены на контейнеровоз, необходимо убедиться, что контейнеры, предназначенные под разгрузку в прямой последовательности, непременно находятся сверху тех, которые будут разгружены в обратной последовательности. Очевидно, что порто-

вая работа могла бы стать более эффективной и менее энергозатратной при применении предварительной сортировки.

В настоящее время исследования о предварительной сортировке контейнеров находятся на начальном этапе. Ким и его команда [1, 2] рассматривали проблему перегруппировки контейнеров. Но они были сосредоточены только на маршруте движения контейнерного крана между различными контейнерными площадками и не касались вопроса о том, как проводить предварительную сортировку контейнеров на отдельно взятой площадке. После этого Ким предложил способ расчета количества контейнеров, подлежащих перегруппировке, оценил эффективность предварительной сортировки и попытался определить точное местоположение перемещенного контейнера с помощью метода ветвей и границ и эвристического подхода.

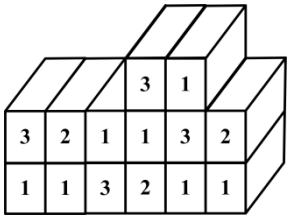
Ли и другие [3, 4] стремились выяснить, как свести к минимуму количество контейнеров, подлежащих предварительной сортировке. Была построена модель независимых пар и оптимизирован маршрут движения эвристическим способом. Родригес-Молинс и его последователи [5] разработали свой метод предварительной сортировки. Но он сулит меньшую производительность и увеличивает ограничивающие условия. В Китае Хао и коллеги [6] построили модель для оптимизации размещения на площадке при случайных условиях на основе поиска фигур и распознавания изображений. Цель – размещение контейнеров без перегруппировки. Бянь и прочие исследователи [7] предложили двухступенчатый гибридный алгоритм в сочетании с поиском соседства и алгоритм независимых пар в контексте рассматриваемой проблемы.

Еще один вариант инновационного алгоритма и оптимизированная модель предварительной сортировки представлены в этой статье.

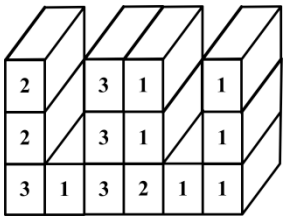
1. ПОСТРОЕНИЕ ОПТИМИЗИРОВАННОЙ МОДЕЛИ

1.1. Подход к решению проблемы

Один контейнерный терминал состоит из нескольких площадок, а задача предварительной сортировки превращается в разработку последовательности перемещения



(a) – Первоначальное расположение



(b) – Окончательное расположение

Рис. 1. Иллюстрация проблемы предварительной сортировки.

контейнеров и перегруппировку на площадке, чтобы убедиться, что контейнеры судна, поставленного под погрузку в первую очередь, находятся сверху контейнеров судна, следующего по очереди. Рис. 1(а) показывает первоначальную схему расположения, а рис. 1(б) – окончательную схему.

Одна гипотеза заключается в том, что все работы по предварительной сортировке проводятся на одной площадке, а другая гипотеза – что контейнеры на ней имеют одинаковый размер и последовательность их загрузки является фиксированной.

Нужно достичь двух целей: во-первых, убедиться, что контейнеры больше не нужно будет перемещать, если они размещены в соответствии с окончательной схемой расположения; во-вторых, время перемещения контейнеров необходимо сократить настолько, насколько это возможно.

1.2. Математическая модель

Переменные и параметры в модели включают в себя:

N: Количество отсеков.

M: Максимальный номер партии.

H: Максимальная высота отсека.

$h_i(0)$: Высота контейнеров в ряду i в начальный момент времени.

$a_{ij}(0)$: Номер партии контейнера j в ряду i в начальный момент времени.

i=1,2,..., N, j=1,2,..., $h_i(0)$;

$$a_{ij}(0)=\begin{cases}k, & k=1,2,\cdots,M,\ 0\le j\le h_i(0),\forall i\\ 0, & \textit{другие}\end{cases}$$

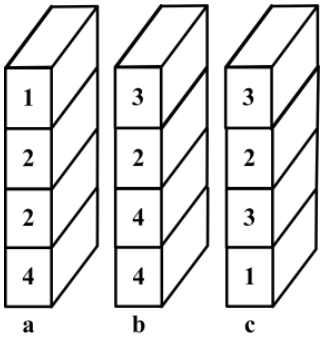


Рис. 2. Пример предварительной сортировки.

Переменные решения:

$x(t)=i$ – означает разгрузку контейнера на вершине ряда j на этапе t;

$y(t)=j$ – означает погрузку контейнера, который выгружается на этапе t на вершине ряда j.

Для предварительной сортировки математическая модель построена следующим образом:

$$\min T \tag{1}$$

при условии, что

$$a_{ij}(T) \geq a_{i,j+1}(T), j = 1,2,\cdots,h_i(T)-1, \forall i \tag{2}$$

$$a_{ij}(t) = \begin{cases} 0, & i = x(t), j = h_i(t-1) \\ a_{x(t),h_{x(t)}(t-1)}(t-1), i = y(t), & j = h_i(t-1)+1 \\ a_{ij}(t-1), & \textit{также} \end{cases} \tag{3}$$

$$h_i(t) = \begin{cases} h_i(t-1)-1, \ i = x(t) \\ h_i(t-1)+1, i = y(t) \\ h_i(t-1), \quad \textit{также} \end{cases} \tag{4}$$

$$h_{x(t)}(t-1) > 0 \tag{5}$$

$$h_{y(t)}(t-1) < H \tag{6}$$

$$x(t) \in \{1,2,\cdots,N\}, t = 1,2,\cdots,T \tag{7}$$

$$y(t) \in \{1,2,\cdots,N\}, t = 1,2,\cdots,T \tag{8}$$

В обозначенной модели формула (1) является целевой функцией для минимального времени предварительной сортировки. Формула (2) – это ограничение для требований окончательной схемы расположения. Оно означает окончание операции по перемещению. То есть все контейнеры в меньшей партии (загружаемые в судно, поставленное под погрузку в первую очередь) находятся сверху контейнеров большей партии (загружаемых во вторую очередь). Формула (3) определяет ограничение для процесса перемещения. Мы предполагаем, что после одного этапа перемещения партия контейнеров в том месте, где находилась перемещенная их часть, составляет 0, а партия перемещенных контейнеров является партией, которой они были. Для других контейнеров партия их нахождения остается неизменной. Формула (4) олицетворяет метод изменения высоты рядов. Число «плюс 1» означает, что один контейнер загружается в ряд, в то время как число «минус 1» значит, что один контейнер выгружается из ряда. Формулы (5) и (6) демонстрируют пределы высоты ряда. Формулы (7) и (8) – ограничения для переменных решения. При этом $x(t)$ и $y(t)$ должны быть целыми числами.

2. РАЗРАБОТКА АЛГОРИТМА

Из-за ее нелинейности математической модели больше подходит надежный метаэвристический алгоритм, нежели традиционный метод математического программирования. Именно поэтому многоступенчатый генетический алгоритм, основанный на двоичном коде, как раз и применен в этой статье к задаче предварительной сортировки.

2.1. Целевая функция

Для такой функции предлагается использовать метод, основанный на числе обратного порядка. Но вначале необходимо ввести некоторые из связанных с ним определений.

Определение 1: В одной последовательности порядок двух фигур противоположен их размеру, две фигуры называются в обратном порядке. Общее число обратного порядка в одной последовательности является обратным числом.

Определение 2: В одном ряду положение (сверху вниз) двух контейнеров противоположно их номеру партии (от меньшей к большей), эти контейнеры называются в обратном порядке. Мы считаем общую сумму обратного порядка, формируемую одним контейнером и контейнерами под ним, числом обратного порядка контейнера. Одно число обратного порядка контейнера умножает расстояние между ним и верхним контейнером и является числом смешанного обратного порядка.

Определение 3: Общая сумма всех чисел путаного обратного порядка контейнера выступает числом путаного обратного порядка ряда. Формула приобретает вид:

Рис. 3. Иллюстрация решения.

$$confuse[i] = \sum_{h=1}^{h(i)} N_{ih} \times d_{ih}, \qquad (9)$$

где $confuse[i]$ – число путаного обратного порядка ряда i; $h(i)$ – высота ряда i; N_{ih} – число в обратном порядке контейнера с высотой h в ряду I; d_{ih} – расстояние между контейнером с высотой i и контейнером наверху. Как показано на рис. 2, числа обратного порядка в рядах a, b и c – это 0, 1 и 7 соответственно.

Определение 4: Общая сумма всех чисел путаного обратного порядка рядов – это число смешанного обратного порядка площадки. Формула приобретает следующий вид:

$$confuse = \sum_{i=1}^{N} confuse[i], \qquad (10)$$

где $confuse$ – это число обратного порядка на конкретной площадке; N – номер ряда в верхней части площадки.

Очевидно, что число обратного порядка показывает, как много трудностей присутствует при предварительной сортировке, и тем самым демонстрирует свое предназначение быть целевой функцией. Цель же алгоритма состоит в уменьшении числа обратного порядка, пока оно не превратится в 0, причем, конечно, посредством предварительной сортировки.

2.2. Метод кодирования

Поскольку максимальное время перемещения не может быть установлено ранее расчета, требуется иметь многоступенчатый период генетического алгоритма с двоичным кодированием (x, y). Например, K = 5, если

X: 65341,

Y: 42135.

Это означает, что операция по перемещению, во-первых, есть перемещение от ряда 6 к ряду 4; во-вторых, от ряда 5 к ряду 2; остальное остается неизменным до шага 5.

В подобном виде бинарного кодирования может возникнуть слишком много ненужных перемещений, если хромосомы X и Y принимают участие в генетическом алгоритме одновременно. Избегая этого, мы вводим X в генетический алгоритм, в то время как Y генерируется из некоего эвристического оператора.

Мы могли бы описать эвристический оператор следующим образом: когда происходит перемещение к верхнему контейнеру в ряду m, уменьшенное значение $E[n]$ числа обратного порядка в других рядах должно быть зафиксировано в первую очередь.

$$E[n] = confuse[n] - confuse'[n], \qquad (11)$$

где $confuse[n]$ – число обратного порядка до перемещения верхнего контейнера в ряду m к ряду n; $confuse'[n]$ – число обратного порядка после перемещения верхнего контейнера в ряду m к ряду n.

2.3. Операции в генетическом алгоритме

Только одна хромосома в каждой части может участвовать в операциях по пересечению и изменению. Тогда используется единая точка метода пересечения контакта. Это означает, что выбор точки контакта случаен, и он меняет подцепочки (части строки) на обеих сторонах выбранной точки соприкосновения. И случайное целое число между 1 и N рассматривается как разрядное значение в генной вариации.

В этом многоступенчатом генетическом алгоритме K шагов работы по перемещению будет сделано в каждой фазе, число обратного порядка станет постоянно снижаться до тех пор, пока не дойдет до 0, алгоритм заканчивается в то же время.

2.4. Решение по усовершенствованию

По окончании алгоритма будет получено решение с T шагами работы по перемещению (рис. 3).

Стоит отметить, что решение может включать некоторые бесполезные операции.

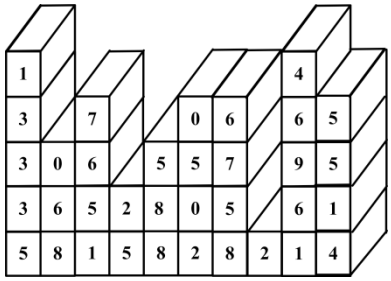


Рис. 4. Первоначальная схема расположения в эксперименте 1.

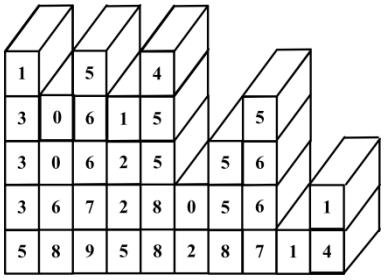


Рис. 5. Финальная схема эксперимента 1.

Таблица 1

Сравнение производительности алгоритма

	Продолжительность работы	Длина шага	Оптимальное решение
Алгоритм, демонстрируемый в статье	63 с	18	(9,4) (7,3) (2,7) (2,7) (8,4) (8,7) (9,7) (5,1) (6,7) (8,9) (6,1) (2,6) (2,3) (9,2) (1,2) (8,2) (7,2) (5,2)
[4]	5 с	31	(8,7) (4,1) (3,7) (2,4) (6,7) (2,4) (2,3) (2,7) (8,4) (0,3) (8,2) (9,6) (1,2) (5,1) (5,2) (9,2) (9,2) (7,3) (9,6) (5,0) (7,9) (8,9) (6,9) (7,5) (7,9) (6,1) (6,3) (6,8) (3,6) (1,6) (8,6)

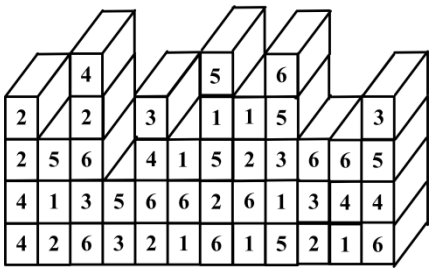


Рис. 6. Первоначальная схема эксперимента 2.

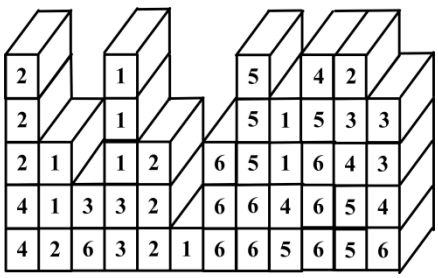


Рис. 7. Окончательная схема эксперимента 2.

Ситуация 1: (a, b) (b, a) → ().

Для того же самого контейнера, когда мы перемещаем его из ряда a в ряд b и тут же из ряда b в ряд a незамедлительно, перемещение (a, b) и перемещение (b, a) нужно отменить.

Ситуация 2: (a, b) (b, c) → (a, c).

Для того же самого контейнера, когда мы перемещаем его из ряда a в ряд b, а затем из ряда b в ряд c незамедлительно, перемещение (a, b) и перемещение (b, c) необходимо объединить для перемещения (a, c).

Ситуация 3: (a, b) (c, d) (b, a) → (c, d).

Для того же самого контейнера, когда мы перемещаем его из ряда a в ряд b, а затем из ряда b в ряд a, если все операции ничего не меняют с рядом a с рядом b, перемещение (a, b) и перемещение (b, a) следует отменить.

Ситуация 4: (a, b) (c, d) (b, e) → (a, e) (c, d).

Для того же самого контейнера, когда мы перемещаем его из ряда a в ряд b, затем из ряда b в ряд e, если производимая работа ничего не делает с рядом a, рядом b и рядом e, перемещение (a, b) и перемещение (b, e) необходимо объединить для перемещения (a, e).

Очевидно, что в ситуациях 1 и 2 описано непрерывное бесполезное перемещение. Прерывистое бесполезное перемещение наблюдается в ситуациях 3 и 4. С целью улучшения решения эвристический оператор используется изначально для отмены бесполезных операций и сохранения схемы расположения контейнеров в секции неизменной.

3. ЭКСПЕРИМЕНТАЛЬНЫЙ АНАЛИЗ

В модельном эксперименте алгоритм выполняется программой Visual C++6.0. Для сравнения все результаты получены на платформе Intel Core i5 1.9GHz CPU.

Таблица 2

Сравнение производительности алгоритма

	Продолжительность работы	Длина шага	Оптимальное решение
Эта статья	335 с	36	(7,3) (9,4) (8,4) (2,7) (8,7) (2,7) (9,3) (9,11) (9,0) (8,9) (3,9) (10,9) (3,9) (2,9) (4,3) (7,3) (10,2) (10,3) (8,10) (6,10) (2,10) (8,10) (7,10) (8,3) (4,8) (6,8) (7,4) (7,8) (4,7) (4,7) (1,7) (6,7) (6,4) (11,6) (11,6) (5,1) (11,7) (6,11) (2,4) (6,11) (5,6) (2,6)
[3]	318 с	47	(3,0) (4,3) (11,3) (4,1) (9,11) (9,3) (5,7) (0,11) (0,9) (0,9) (2,0) (2,9) (10,2) (10,0) (7,10) (7,3) (7,9) (7,0) (3,7) (6,10) (6,7) (6,3) (6,5) (4,6) (5,4) (2,6) (8,6) (2,6) (11,2) (11,8) (11,4) (0,4) (11,0) (5,11) (8,11) (1,11) (2,5) (2,11) (4,2) (1,2) (3,2) (8,2) (8,3) (8,1) (5,8) (10,8) (4,8)

Первоначальная схема расположения контейнеров в эксперименте 1 такая же, как и в [4] (на площадке 10 рядов, в каждом из них в высоту размещены 5 контейнеров). Схема показана на рис. 4.

С помощью алгоритма авторы сравнивали оптимальное по целям решение с решениями в [4]. Результаты представлены в таблице 1.

По ходу анализа обнаружено, что, хотя время работы по алгоритму, представленному в статье, больше (63 с – приемлемое время), но качество решения лучше и длина шага – это только половина шага в [4]. Все операции по предварительной сортировке завершаются в течение всего лишь 18 шагов, и окончательная схема расположения в эксперименте 1 показана на рис. 5. Предварительная сортировка по описанному алгоритму обещает большую экономическую выгоду.

Первоначальная схема размещения контейнеров в эксперименте 2 такая же, что и описанная в [3] (на площадке 12 рядов высотой в 5 контейнеров). Схема показана на рис. 6.

С помощью алгоритма сравнивается оптимальное решение с решениями в [3]. Результаты в таблице 2.

Из анализа следует, что результаты с целочисленным программированием на основе сетевой модели потока в [3] и результаты эксперимента свидетельствуют о преимуществе предложенного в статье алгоритма: время работы по его схеме всего 335 с, но качество решения лучше. Все операции по предварительной сортировке выполняются за 36 шагов, на 11 шагов меньше, чем с алгоритмом в [3]. Окончательная схема эксперимента 2 представлена на рис. 7.

ВЫВОДЫ

В проведенном исследовании рассмотрена проблема предварительной сортировки контейнеров на терминальных площадках морских портов. Была построена математическая модель программирования, базовым для которой стал многоступенчатый генетический алгоритм, основанный на бинарном кодировании. В этом алгоритме авторы дают определение числу обратного порядка, делают его целевой функцией и предлагают использовать возможности эвристического оператора при выработке решений. Результаты экспериментов показали, что алгоритм, демонстрируемый в журнальной статье, способен обеспечить более оптимальное смещение длины шага, чем алгоритмы, построенные на эвристических методах.

ЛИТЕРАТУРА

1. Kim, K. H., Bae, J. W. Re-marshalling export containers in port container terminals. *Computers and Industrial Engineering*, 1998, 35, pp. 655–658.
2. Kim, K. H., Hong, G. P. A heuristic rule for relocating blocks. *Computers and Operations Research*, 2006, 33 (4), pp. 940–954.
3. Lee, Y., Hsu, N. Y. An optimization model for the container pre-marshalling problem. *Computers and Operations Research*, 2007, 34, pp. 3295–3313.
4. Lee, Y., Chao, S. H. A neighborhood search heuristic for pre-marshalling export containers. *European Journal of Operational Research*, 2009, 196 (2), pp. 468–475.
5. Rodriguez-Molins, M., Salido, M. A., Barber, F. Intelligent planning for allocating containers in maritime terminals. *Expert Systems with Applications*, 2012, 39(1), pp. 978–989.
6. Hao Jumin, Ji Zhuoshang, Lin Yan. Optimization model in mixture order contain yard. *Journal of Dalian University of Technology*, 2000, 40(1), pp. 102–105.
7. Bian Zhan, Li Na, Li Xiangjun. Mixture optimization algorithm of pre-marshaling problem in container yard. *Control and decision*, 2014, 29(2), pp. 373–378. ●

Координаты авторов: **Ли Хаоюань** – lihaoyuan@neusoft.edu.cn, **Сан Донгши** – sundongshi@neusoft.edu.cn.

Статья поступила в редакцию 01.06.2015, принята к публикации 24.08.2015.

MULTI-STAGE GENETIC ALGORITHM FOR CONTAINER PRE-MARSHALLING PROBLEM

Li Haoyuan, *Neusoft University of Information, Dalian, People's Republic of China.*
Sun Dongshi, *Neusoft University of Information, Dalian, People's Republic of China.*

ABSTRACT

In order to solve the pre-marshalling problem in container yard, a mathematical programming model and a multi-stage genetic algorithm with binary encoding scheme are required. In the proposed algorithm, the objective function is designed via the definition of confuse coefficient; several heuristic operators provide development of necessary methodical and operational actions. The way how to achieve better solutions is shown at the example of two researches, confirming the efficiency of genetic algorithm with binary encoding as an optimizer to construct a container pre-marshalling model.

Keywords: sea transportation, terminal, container, pre-marshalling, genetic algorithm, binary encoding, mathematical model, heuristics.

Background. *In general, normal export containers are delivered to container terminal in a random sequence within seven days before ship departure. If the containers are loaded on container ship in the order they were located in the terminal, it will very likely lead to a big problem. When a container ship arrives, it may be found that the containers belonging to this ship are under some containers belonging to other ship. In that situation, all the other ships' containers need to be regrouped and reloaded, that will result in time expenditures.*

In order to solve this problem, the term «pre-marshalling» is put forward in this article. Pre-marshalling means that containers need to be regrouped in advance. Thus before they are loaded into the container ship it is necessary to be sure that containers which need to be unloaded in forward sequence are always on the top of those ones in backward sequence. Obviously, the port work could be more efficient and less energy-consuming through pre-marshalling.

At present, the research about container pre-marshalling is at the initial phase. Kim and his team [1, 2] have considered the container regrouping problem. But they focused on moving route of the container crane between different bays, and not solved an issue how to pre-marshal the container in one specific bay. After that, Kim put forward a technique to calculate the number of containers to be regrouped, evaluated the performance of pre-marshalling, and tried to find an exact location of shifted container by using branch-bound method and heuristic approach.

Lee and others [3, 4] sought to find out how to minimize the number of pre-marshalled containers. An independent pairs (IP) model was built and the moving route was optimized by heuristic way. Rodriguez-Molins and his followers [5] designed a specific method for pre-marshalling. But the method could show less performance even increases number of limiting conditions. In China, Hao and colleagues [6] built a model to optimize the bay location under random conditions on the base of figure search and pattern recognition. His goal was to arrange containers without regrouping. Bian and other researchers [7] offered a two stage hybrid algorithm combining neighborhood search and IP algorithm in the context of considered issue.

An innovative algorithm and optimized pre-marshalling model are highlighted in this article.

Objective. *The objective of the authors is to consider multi-stage genetic algorithm applicable to solve a problem of container pre-marshalling at terminals of sea ports.*

Methods. *The authors use general scientific methods, mathematical programming, multi-stage genetic algorithm, modeling, simulation, hypothesis construction, comparative analysis.*

Results.

1. Construction of a mathematical model

1.1. Problem description

One container yard consists of several bays, and pre-marshalling problem turns out to design the container shifting sequence and regrouping in the bay to make sure the forward-ship containers locate on the top of the backward-ship ones. Pic. 1 (a) shows an initial layout in a bay, and Pic. 1 (b) is a final layout in a bay.

(a) – Initial container layout in a bay ***(b) – Final layout in a bay***

Pic. 1. Illustration of pre-marshalling problem.

One hypothesis is that all the pre-marshalling operations take place in one bay and another hypothesis is that the containers in one bay have the same size, and the loading sequence is fixed.

Two goals should be achieved: firstly, to make sure that there will be no further shifting of containers if they are loaded in the sequence of the final layout; secondly, it is necessary to reduce shifting time as much as possible.

1.2. Mathematical model

Variables and parameters in the model include:

N: The number of bays

M: Maximum batch number

H: Maximum height of the bay

$h_i(0)$: The height of containers in row i at the initial time.

$a_{ij}(0)$: The batch number of container j in row i at the initial time.

i=1,2,..., N, j=1,2,..., $h_i(0)$

$$a_{ij}(0)=\begin{cases} k, & k=1,2,\cdots,M,\ 0\le j\le h_i(0),\forall i \\ 0, & else \end{cases}$$

Decision variable:

x(t)=i means unloading of container on the top of row j on step t;

y(t)=j means loading of container which is unloaded on step t on the top of row j.

For pre-marshalling a mathematical model is built as follows:

$$\min T \qquad (1)$$

such that

$$a_{ij}(T)\ge a_{i,j+1}(T), j=1,2,\cdots,h_i(T)-1,\forall i \qquad (2)$$

$$a_{ij}(t)=\begin{cases} 0, & i=x(t), j=h_i(t-1) \\ a_{x(t),h_{x(t)}(t-1)}(t-1), & i=y(t),\quad j=h_i(t-1)+1 \\ a_{ij}(t-1), & else \end{cases} \qquad (3)$$

$$h_i(t)=\begin{cases} h_i(t-1)-1, & i=x(t) \\ h_i(t-1)+1, & i=y(t) \\ h_i(t-1), & else \end{cases} \qquad (4)$$

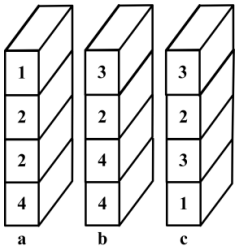
$$h_{x(t)}(t-1)>0 \qquad (5)$$

$$h_{y(t)}(t-1)<H \qquad (6)$$

$$x(t)\in\{1,2,\cdots,N\}, t=1,2,\cdots,T \qquad (7)$$

$$y(t)\in\{1,2,\cdots,N\}, t=1,2,\cdots,T \qquad (8)$$

In the described model, formula (1) is the objective function for minimum pre-marshalling time. Formula (2) is the constraint for the requirements of the final layout. It stands for the end of shifting work. That is, all the containers in smaller batch (loaded in forward ship) are on the top of the ones in bigger batch (loaded in backward ship). Formula (3) determines the constraint for the shifting operation process. We assume that after one step of shifting, the batch of the containers in the locaion, where the shifted containers were, is 0, and the batch of the shifted containers is the batch where they were in. For other containers, the batch stays the same. Formula (4) is changing method for the height of rows. The figure «plus 1» means that one container is loaded in the row, while the figure «minus 1» means that one container is unloaded from the row. Formula (5) and formula (6) are the height limits of row. Formula (7) and formula (8) are the constraints for decision variables. Expressions x(t) and y(t) must be integer.



Pic. 2. An example of pre-marshalling.

2. Algorithm design

Due to the nonlinearity of the mathematical model, the robust meta-heuristic algorithm is more suitable than the traditional mathematical programming method. So, a multi-stage genetic algorithm based on binary code is applied to the pre-marshalling problem in this article.

2.1. Objective function

For this function it is proposed to use a method based on the number of reverse order. At first, some related definition need to be introduced.

Definition 1: In one sequence, the order of two figures is opposite to their size, two figures are called in a reversed order. The total number of reversed order in one sequence is the inverse number.

Definition 2: In one row, the position (from top to bottom) of two containers is opposite to their batch number (from small to big), these containers are called in a reversed order. We call the total sum of reverse order forming by one container and the containers below it the container's reverse order number. One container's reverse order number multiplies the distance between it and the top container and is the container's confused reverse order number.

Definition 3: The sum total of all of containers' confused reverse order number is the row's confused reverse order number. The formula is as below:

$$confuse[i]=\sum_{h=1}^{h(i)} N_{ih}\times d_{ih}\ , \qquad (9)$$

where confuse[i] is the row i's confused reverse order number; h(i) is the height of row i; N_{ih} is the number in reverse order of the container with height h in row I; d_{ih} is the distance between the container with height i and the container on the top. As shown in Pic. 2, the reverse order number in rows a, b and c are 0, 1 and 7 respectively.

Definition 4: The sum total of all of rows' confused reverse order number is the bay's confused reverse order number. The formula is as below:

$$confuse=\sum_{i=1}^{N} confuse[i]\ , \qquad (10)$$

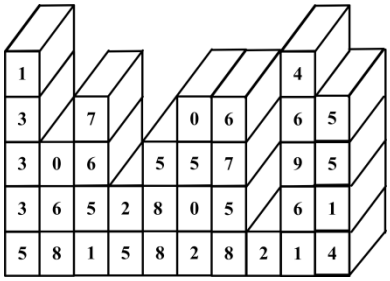
where confuse is the reverse order number in specific bay, N is the row number in the top bay.

Obviously, the reverse order number means how much difficult pre-marshalling operation is, so it is designed to be objective function. And the goal of the algorithm is decreasing the reverse order number until it turns to be 0, of course, through pre-marshalling.

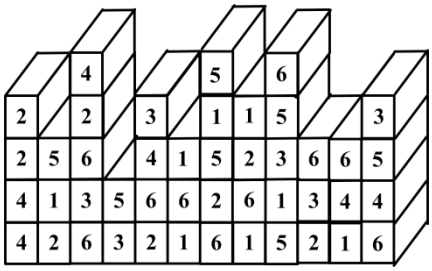
2.2. Coding method

As the maximum time of shifting could not be fixed before calculation, it is necessary to have a multistage period of genetic algorithm with (x, y) binary encoding. For example, K = 5, if

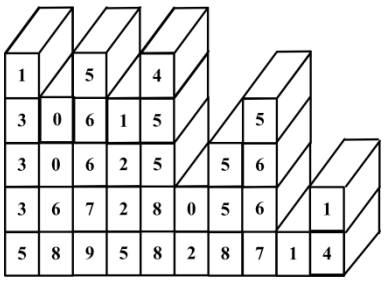
Pic. 3. Illustration of a solution.



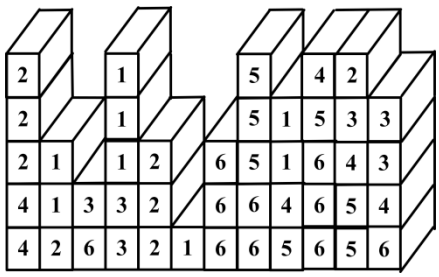
Pic.4. Initial layout of experiment 1.



Pic.5. Final layout of experiment 1.



Pic.6. Initial layout of experiment 2.



Pic.7. Final layout of experiment 2.

X: 65341,
Y: 42135.

That means the shifting operation is: firstly, shifting from row 6 to row 4; secondly, from row 5 to row 2; the rest stays similar until step 5.

In this kind of binary encoding, too much needless shifting may occur if chromosomes X and Y participate in genetic algorithm at the same time. Avoiding this, we enter X into genetic algorithm, while Y is generated from a kind of heuristic operator.

We could describe the heuristic operator as follows: when the shifting to the top container in row m occurs, the decreased value E[n] of reverse order number in other rows needs to be fixed at first.

$$E[n] = confuse[n] - confuse'[n], \quad (11)$$

where confuse[n] is the reverse order number before shifting of the top container in row m to row n; confuse' [n] is the reverse order number after shifting of the top container in row m to row [n].

2.3. Operations in genetic algorithm

Only one chromosome in every part could participate in crossing and variation operations. In this case, single point of contact crossing method is used. That means that choosing a point of contact is random, and it changes the substrings (parts of rows) on both sides of the chosen point of contact. And the random integer between 1 and N is regarded as a place value in genic variation.

In this multi-stage genetic algorithm, K steps of shifting work will be done in every phase, the reverse order number will constantly go down until it comes to be 0, the algorithm ends at the same time.

2.4. Improvement solution

When the algorithm ends, a solution with T steps of shifting work will be obtained (Pic. 3).

It is worth noting that the solution may include some useless operations as follow:

Situation 1: (a, b) (b, a) → ( ).

Table 1

Comparison of algorithm performance

	Operation time	Step length	Optimal solution
The algorithm in this article	63 s	18	(9,4) (7,3) (2,7) (2,7) (8,4) (8,7) (9,7) (5,1) (6,7) (8,9) (6,1) (2,6) (2,3) (9,2) (1,2) (8,2) (7,2) (5,2)
[4]	5 s	31	(8,7) (4,1) (3,7) (2,4) (6,7) (2,4) (2,3) (2,7) (8,4) (0,3) (8,2) (9,6) (1,2) (5,1) (5,2) (9,2) (9,2) (7,3) (9,6) (5,0) (7,9) (8,9) (6,9) (7,5) (7,9) (6,1) (6,3) (6,8) (3,6) (1,6) (8,6)

Table 2

Comparison of algorithm performance

	Operation time	Step length	Optimal solution
This article	335 s	36	(7,3) (9,4) (8,4) (2,7) (8,7) (2,7) (9,3) (9,11) (9,0) (8,9) (3,9) (10,9) (3,9) (2,9) (4,3) (7,3) (10,2) (10,3) (8,10) (6,10) (2,10) (8,10) (7,10) (8,3) (4,8) (6,8) (7,4) (7,8) (4,7) (4,7) (1,7) (6,7) (6,4) (11,6) (11,6) (5,1) (11,7) (6,11) (2,4) (6,11) (5,6) (2,6)
[3]	318 s	47	(3,0) (4,3) (11,3) (4,1) (9,11) (9,3) (5,7) (0,11) (0,9) (0,9) (2,0) (2,9) (10,2) (10,0) (7,10) (7,3) (7,9) (7,0) (3,7) (6,10) (6,7) (6,3) (6,5) (4,6) (5,4) (2,6) (8,6) (2,6) (11,2) (11,8) (11,4) (0,4) (11,0) (5,11) (8,11) (1,11) (2,5) (2,11) (4,2) (1,2) (3,2) (8,2) (8,3) (8,1) (5,8) (10,8) (4,8)

For the same container, when we move it from row a to row b, and move it from row b to row a immediately, then displacement (a, b) and displacement (b, a) need to be canceled.

Situation 2: (a, b) (b, c) → (a, c).

For the same container, when we move it from row a to row b, and then move it from row b to row c immediately, displacement (a, b) and displacement (b, c) need to be combined to move (a, c).

Situation 3: (a, b) (c, d) (b, a) → (c, d).

For the same container, we move it from row a to row b, then move it from row b to row a. If all operations do nothing with row a and row b, displacement (a, b) and displacement (b, a) need to be canceled.

Situation 4: (a, b) (c, d) (b, e) → (a, e) (c, d).

For the same container, we move it from row a to row b, then move it from row b to row e. If all work do nothing with row a, row b and row e, displacement (a, b) and displacement (b, e) need to be combined to move (a, e).

Obviously, continuous useless shifting is described in situations 1 and 2. Discontinuous useless shifting is described in situations 3 and 4. For the improvement of solution, a kind of heuristic operator is put forward to remove useless operation on the premise of keeping the bay layout unchanged.

3. Experimental analysis

In simulation experiment, the algorithm is completed with Visual C++ 6.0. For comparison's sake, all the results are obtained on the platform of Intel Core i5 1.9GHz CPU.

Initial layout of containers in experiment 1 is the same as in [4] (10 rows are included in the bay, and the height of each row is 5). The layout is shown in Pic. 4.

With the help of the algorithm, the authors compared the optimum solution with the solutions in [4]. The results are shown in Table 1.

During the analysis it was found out that though the operation time with the suggested algorithm is longer (though 63 s is an acceptable time), but the quality of solution is better and the step length is just half of that in [4]. All pre-marshalling operations finish within only 18 steps, and the finial layout of experiment 1 is shown in Pic. 5. The pre-marshalling will be completed more quickly with the procedure, described in this algorithm, and more economic benefit will also be obtained.

Initial layout in experiment 2 is the same as in [3]: 12 rows are included in the bay, and the height of each row is 5. The layout is shown in Pic. 6.

Through the algorithm, the optimum solution is compared with the solutions in [3]. The results are shown in Table 2.

The analysis of the results with integer programming based on network flow model in [3], and experiment's results proves the advantage of the algorithm suggested in this article: its operation time according to the scheme is just 335 s, but the quality of solution is better. All pre-marshalling operations finish only within 36 steps, 11 steps less than in the algorithm in [3]. And the finial layout of experiment 2 is shown in Pic. 7.

Conclusions. *Pre-marshalling problem regarding containers at terminal sites of sea ports was considered in the article. A mathematical programming model was built and multi-stage genetic algorithm based on binary coding was used as basic for this model. In this algorithm, the authors define the reverse order number, make it an objective function and offer to use capacities of the heuristic operator for solution evolution. The experiment results have shown under the same simulation environment and conditions, that the suggested algorithm could provide more optimal shifting of step length then algorithms in based on heuristic models.*

REFERENCES

1. Kim, K. H., Bae, J. W. Re-marshalling export containers in port container terminals. *Computers and Industrial Engineering*, 1998, 35, pp. 655–658.

2. Kim, K. H., Hong, G. P. A heuristic rule for relocating blocks. *Computers and Operations Research*, 2006, 33 (4), pp. 940–954.

3. Lee, Y., Hsu, N. Y. An optimization model for the container pre-marshalling problem. *Computers and Operations Research*, 2007, 34, pp. 3295–3313.

4. Lee, Y., Chao, S. H. A neighborhood search heuristic for pre-marshalling export containers. *European Journal of Operational Research*, 2009, 196 (2), pp. 468–475.

5. Rodriguez-Molins, M., Salido, M. A., Barber, F. Intelligent planning for allocating containers in maritime terminals. *Expert Systems with Applications*, 2012, 39(1), pp. 978–989.

6. Hao Jumin, Ji Zhuoshang, Lin Yan. Optimization model in mixture order contain yard. *Journal of Dalian University of Technology*, 2000, 40(1), pp. 102–105.

7. Bian Zhan, Li Na, Li Xiangjun. Mixture optimization algorithm of pre-marshaling problem in container yard. *Control and decision*, 2014, 29(2), pp. 373–378.

Information about the authors:

Li Haoyuan – D.Sc., associate professor, director of the department of business management, Neusoft University of Information, Dalian, China, lihaoyuan@neusoft.edu.cn.

Sun Dongshi – Master Degree, lecturer, logistics team leader, Neusoft University of Information, Dalian, China, sundongshi@neusoft.edu.cn.

Article received 01.06.2015, accepted 24.08.2015.