I.J. Intelligent Systems and Applications, **2014, 03, 1-11**
Published Online February 2014 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2014.03.01

Energy Resource Management of Assembly Line Balancing Problem using Modified Current Search Method

Supaporn Suwannarongsri
Rattanakosin College for Sustainable Energy and Environment (RCSEE),
Rajamangala University of Technology Rattanakosin, Nakhon Pathom, Thailand
E-mail: suwannarongsri@hotmail.com

Tika Bunnag, Dr., **Waraporn Klinbun**, Dr.
Rattanakosin College for Sustainable Energy and Environment (RCSEE),
Rajamangala University of Technology Rattanakosin, Nakhon Pathom, Thailand
E-mail: tbunnag@hotmail.com; tukatuk4@gmail.com

***Abstract*—** This paper aims to apply a modified current search method, adaptive current search (ACS), for assembly line balancing problems. The ACS algorithm possesses the memory list (ML) to escape from local entrapment and the adaptive radius (AR) mechanism to speed up the search process. The ACS is tested against five benchmark unconstrained and three constrained optimization problems compared with genetic algorithm (GA), tabu search (TS) and current search (CS). As results, the ACS outperforms other algorithms and provides superior results. The ACS is used to address the number of tasks assigned for each workstation, while the heuristic sequencing (HS) technique is conducted to assign the sequence of tasks for each workstation according to precedence constraints. The workload variance and the idle time are performed as the multiple-objective functions. The proposed approach is tested against four benchmark ALB problems compared with the GA, TS and CS. As results, the ACS associated with the HS technique is capable of producing solutions superior to other techniques. In addition, the ACS is an alternative potential algorithm to solve other optimization problems.

***Index Terms*—** Metaheuristics, Adaptive Current Search, Tabu Search, Assembly Line Balancing, Energy Resource Management

I. Introduction

Conventionally, effective energy resource management consists of energy management program, organization structure, energy policy, planning, auditing and reporting [1]. Moreover, classical method for energy resource management uses the various principles such as energy management system (EMS), excess energy management (EEM), energy driven force (EDF), energy knowledge management (EKM), energy innovation creation (EIC), energy system control (ESC) and energy organization culture (EOC) [2]. The conventional orientation focuses on operations and activities done by human. However, due to human uncertainty in operation, those approaches seem inefficient and non-sustainable. Generally, energy resource management conducted recommendations of consultants usually based on their experience and feeling. The motivation of this research is initiated from needs to find the general algorithm to solve the energy resource management in industries efficiently.

Regarding to optimization context, many of challenging applications in science, engineering and technology can be formulated and performed as optimization problems [3,4]. Energy management can be considered as a class of non-polynomial time hard (NP-hard) combinatorial optimization problem. Such the problem usually possesses nonlinear and unsymmetrical terms as well as multiple local solutions. These cause the problem complex and difficult to solve by an exact method within a reasonable amount of time. Moreover, inefficient algorithms are easily trapped due to its local solutions. To solve the problem, metaheuristic optimization methods are alternatives [4].

Over five decades, many efficient metaheuristics such as genetic algorithm (GA), tabu search (TS), particle swarm optimization (PSO), harmony search (HS) and current search (CS) have been developed for combinatorial, continuous and multiobjective optimization problems [5]. These algorithms can be classified into two groups: population-based and single-solution based algorithms. Two important properties of any metaheuristics are exploration and exploitation [6-8]. Exploration property simply means the generation of diverse solutions to explore the search space on the global scale, while exploitation focuses on the search in a local region by exploiting information that a currently good solution is found in a local region. It was found by literatures that the population-based has strong exploration property, while the single-solution based has strong exploitation property. Metaheuristics has

been widely applied to various real-world engineering problems [5,7]. One of the latest metahueristic search algorithms is the current search (CS) [9]. By literatures, the CS has been successfully applied to control engineering [10] and signal processing [11]. The CS performed satisfactory results. However, the CS may be trapped by local optima. Therefore, the CS algorithms need to be modified to improve its effectiveness.

This paper proposes the adaptive current search (ACS), which is the modified version of the CS method as a new tool for solving the resource management optimization problems. It consists of five sections. The ACS algorithms are elaborately explained in section II. The performance evaluation of the ACS over five benchmark unconstrained and three constrained optimization problems compared with GA, TS and CS is provided in section III. The application of the ACS to energy resource management of the assembly line balancing (ALB) problems is illustrated in section IV, while conclusions are given in section V.

II. Adaptive Current Search Algorithms

The current search (CS) has been developed based on the principle of current divider in electric circuit theory [9,10,11]. The behavior of electric current is like a tide that always flow to lower places. The less the resistance of branch, the more the current flows. The CS algorithm is described by the flow diagram represented in Figure 1. The advantages are that the CS can find any local solution in each search direction efficiently. Moreover, it can provide unlimited search directions defined by users. However, the CS algorithm is based on an electric current behavior considered as the natural movement. The disadvantages of the CS are that its search process may be trapped or locked by any local solution. In addition, the search time consumed by the CS is depended on the numbers of search directions.

The modified version of the CS algorithm is called the adaptive current search (ACS) metahueristics. The memory list (ML) and the adaptive radius (AR) mechanism are proposed for inserting into the CS algorithms. The ML is used to escape from local entrapment caused by any local solution, while the AR is conducted to speed up the search process. The proposed ML consists of three levels: low, medium and high. The low-level ML is used to store the ranked initial solutions at the beginning of search process, the medium-level ML is conducted to store the solution found along each search direction, and the high-level ML is used to store all local solutions found at the end of each search direction.

Once the search process is started, the set of initial solutions is generated randomly. They will be ranked by the objective function of the problem of interest from most to least significant. The ranked initial solutions will be stored in the low-level ML. This scheme helps the ACS algorithms to be stronger in exploration strategy.

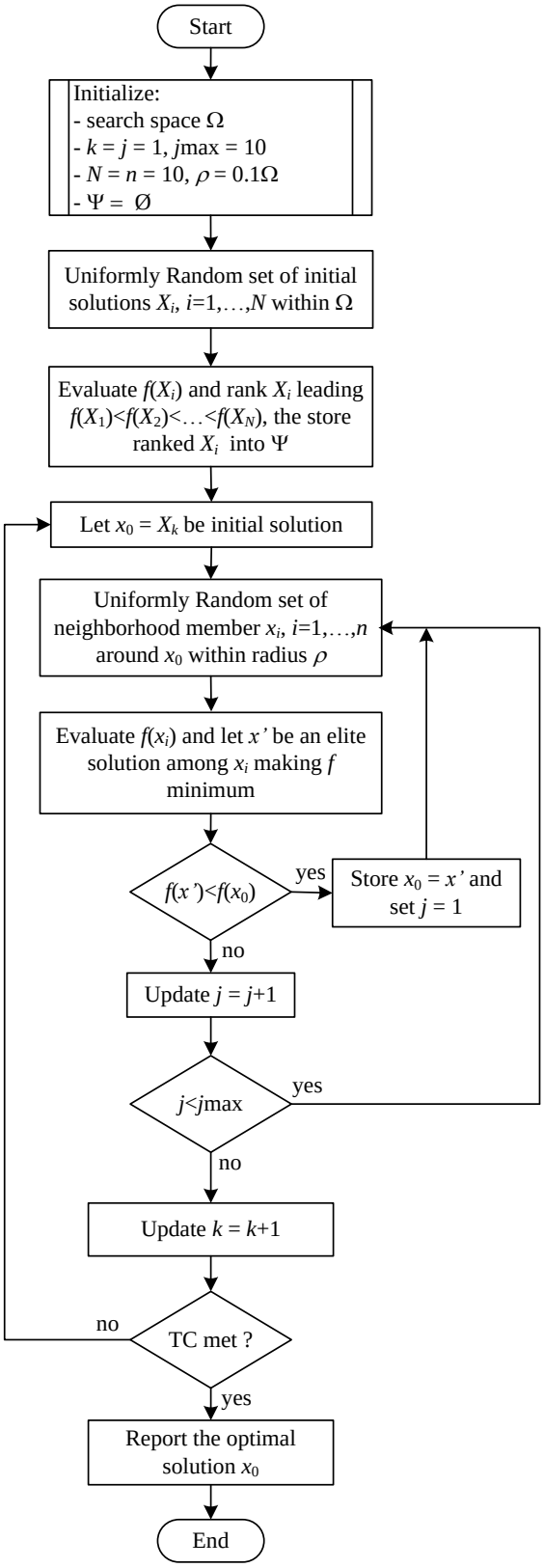


Fig. 1: Flow Diagram Of CS Algorithm [9]

The ranked initial solution with most significant will be selected as the current solution for the first search direction. The neighborhood members will be generated randomly around the current solution within the certain radius. All neighborhood members will be evaluated by

the objective function of the problem. The best neighborhood giving the minimum objective function value is used to compare with the current solution. If the best neighborhood is better than the current solution, the current solution will be replaced by the best neighborhood, while the previous current solution will be keep into the medium-level ML. Once the search process moves close to the local solution (which can be observed by the objective function value), the AR mechanism is activated to speed up the process. The search radius is thus reduced according to the objective function value found so far. The less the objective function value found, the smaller the search radius is adapted. This implies that the refined solutions will be considered. Then, the local solution of this search direction will be found within faster time and will be keep into the high-level ML. This scheme helps the ACS algorithms to be stronger in exploitation strategy.

After the local solution of the first search direction is found, the search process look backward to the low-level ML. The second most initial solution ranked in the low-level ML will be selected as the current solution for the second search direction so far. This scheme will be repeated until the global solution will be found or the termination criteria are met.

With ML and AR mechanism, a sequence of solutions obtained by the ACS rapidly converges to the global minimum. The developed ACS algorithms can be described step-by-step as follows:

Step 1 Initialize the search space Ω, iteration counter $k = j = 1$, maximum allowance of solution cycling $j_{\max}$, number of initial solutions N, number of neighborhood members n, search radius R, and low-level ML $\Psi = \varnothing$, medium-level ML $\Gamma = \varnothing$, and high-level ML $\Xi = \varnothing$.

Step 2 Uniformly random initial solution X_i, $i = 1,\ldots,N$ within Ω.

Step 3 Evaluate the objective function $f(X_i)$ for $\forall X$. Rank X_i that gives $f(X_1) < \cdots < f(X_N)$, then store ranked X_i into the low-level ML Ψ.

Step 4 Let $x_0 = x_k$ as selected initial solution.

Step 5 Uniformly random neighborhood x_i, $i = 1,\ldots,n$ around x_0 within radius R.

Step 6 Evaluate the objective function $f(x_i)$ for $\forall x$. A solution giving the minimum objective function is set as x^*.

Step 7 If $f(x^*)<f(x_0)$, keep x_0 into medium-level ML Γ_k and set $x_0 = x^*$, set $j = 1$ and return to Step 5. Otherwise keep x^* into medium-level ML Γ_k and update $j = j+1$.

Step 8 If the search process moves close to the local solution, activate the AR mechanism by adjusting $R = \rho R$, $0<\rho<1$.

Step 9 If $j< j_{\max}$, return to Step 5. Otherwise keep x_0 into high-level ML Ξ and update $k = k+1$.

Step 10 Terminate the search process when the termination criteria (TC) are satisfied. The optimum solution found is x_0. Otherwise return to Step 4.

ACS algorithms mentioned above can be represented by the pseudocode as shown in Figure 2, and some movements of the ACS search process can be depicted in Figure 3.

Pseudo-code of ACS

```
procedure
    Initialize the search space = Ω,
    the memory lists (ML) Ψ = Γ = Ξ = ∅;
    the iteration counter i = j = k = l = 1,
    the maximum search iteration in each direction I_max,
    the maximum allowance of solution
    cycling j_max, the number of initial solutions
    (feasible directions of currents in network) N,
    number of neighborhood members n,
    the maximum objective function value ε for AR,
    and the search radius R.
    while (i < N) do // generate search directions
        Generate initial solutions X_i randomly
        within Ω; Evaluate f(X_i) via the objective
        function; Rank X_i giving f(X_i) from min
        to max values (X_1 gives the min, while
        X_N gives the max objective function values);
        Store ranked X_i into Ψ;
    end_while
    while (k ≤ N) or the TC are not satisfied do
        Set x_0 = X_k;
        while (i ≤ I_max) do
            Generate neighborhood x_{k,l} (l =1,2,…,n)
            randomly around x_0 within R;
            for l ←1 to n do
                Evaluate f(x_{k,l}) via the objective
                function; Set x* as a solution giving
                the minimum objective value;
            end_for
            if f(x*) < f(x_0) do
                Keep x_0 into Γ_k;
                Set x_0 = x*;
                Set j = 1;
            else do
                Keep x* into Γ_k;
                Update j = j+1;
            end_if
            if f(x_0) < ε do // activate the AR
                Adapt R = ρR, ρ ∈ [0, 1];
            end_if
            if j == j_max do // activate other directions
                Keep x_0 into Ξ;
                Update k = k+1;
                Go out-loop while;
            end_if
            Update i = i+1;
        end_while
        Keep x_0 into Ξ;
        Update k = k+1;
    end_while
end_procedure
```

Fig. 2: Pseudocode Of ACS Algorithm

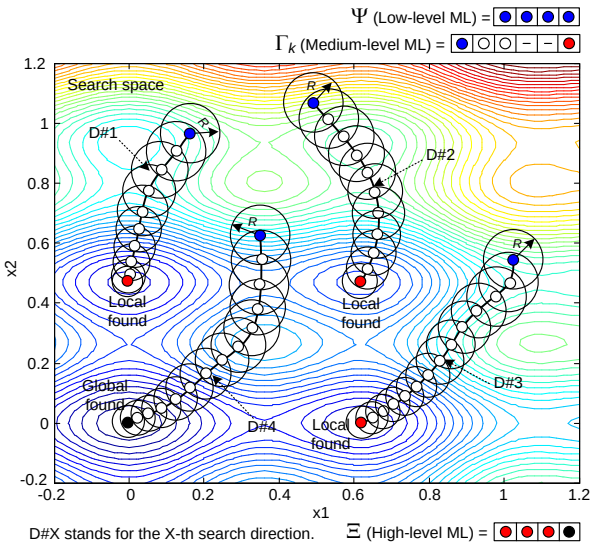


Fig. 3: Some Movements Of ACS Algorithm

III. Performance Evaluation

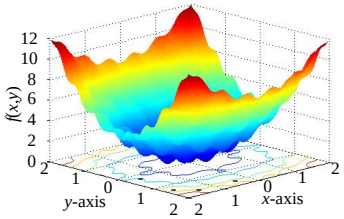
This section presents the performance evaluation of the ACS against unconstraint and constraint benchmark optimization problems compared with GA, TS and CS. The efficiency of those algorithms can be measured by the success rate of finding the global optima. In this paper, algorithms of GA and TS are omitted. Readers may refer to [12,13] for GA and [14,15] for TS, respectively. Those algorithms were coded by MATLAB running on Intel Core2 Duo 2.0 GHz 3 Gbytes DDR-RAM computer, while GA is used from the MATLAB-GA Toolbox [13]. Each algorithm performs search on each test function for 100 trials beginning with different initial solutions while search parameters are kept the same. Search parameter settings for the GA follow MATLAB-GA Toolbox [13] and for the TS follow [15].

3.1 Unconstrainted Optimization

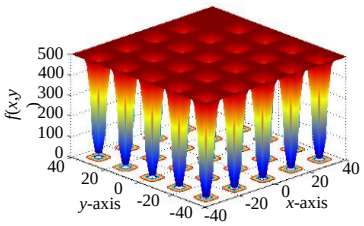
For unconstraint optimization test, GA, TS, CS and ACS are tested against five well-known unconstrained optimization problems [16] including Bohachevsky function (BF), the fifth function of De Jong (DJF), Griewank function (GF), Michaelwicz function (MF) and Shekel's fox-holes function (SF). These problems are summarized in Table 1, in which J_{min} is the minimum values of objective functions required to terminate the search. The search parameters of the CS and the ACS for this test are set as summarized in Table 2, where n = number of neighborhood members, R = search radius, N = search (current) paths and I_{max} = maximum search iterations.

Table 1: Summary Of Unconstrained Optimization Problems

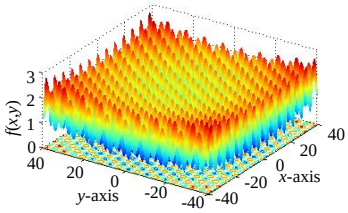
BF: $f(x,y) = x^2 + 2y^2 - 0.3\cos(3\pi x) - 0.4\cos(4\pi y) + 0.7$
global minimum located at $x = 0$, $y = 0$ with $f(x,y) = 0$,
search space: $x,y \in [-2, 2]$ and $J_{min} \le 1 \times 10^{-6}$



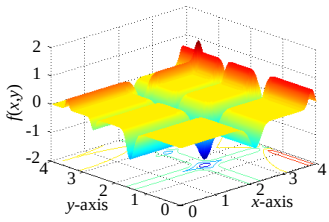
DJF: $f(x,y) = [1/500 + \sum_{j=1}^{25} \{1/(j + (x - a_{1j})^6 + (y - a_{2j})^6)\}]^{-1}$,
where $a_{ij} = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \cdots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \cdots & 32 & 32 & 32 \end{pmatrix}$
global minimum located at $x = -32$, $y = -32$ with $f(x,y) = 0.9980$,
search space: $x,y \in [-40, 40]$ and $J_{min} \le 0.9990$



GF: $f(x,y) = 1 + \{(1/4000)(x^2 + y^2)\} - \{\cos(x)\cos(y/\sqrt{2})\}$
global minimum located at $x = 0$, $y = 0$ with $f(x,y) = 0$,
search space: $x,y \in [-40, 40]$ and $J_{min} \le 1 \times 10^{-6}$



MF: $f(x,y) = -\sin(x)\sin^{20}(x^2/\pi) - \sin(y)\sin^{20}(2y^2/\pi)$
global minimum located at x =2.20319, y = 1.57049 with
$f(x,y)$ = -1.8013, search space: $x,y \in [0, 4]$ and $J_{min} \le$ -1.80129



SF: $f(x,y) = -\sum_{j=1}^{30} [1/\{c_j + (x - a_{1j})^2 + (y - a_{2j})^2\}]$,
where $a_{ij} = \begin{pmatrix} 9.6810 & 9.4000 & 8.0250 & \cdots & 9.4960 & 4.1380 \\ 0.6670 & 2.0410 & 9.1520 & \cdots & 4.8300 & 2.5620 \end{pmatrix}$,
$c_j = (0.8060\ \ 0.5170\ \ 0.1000\ \ 0.9080\ \cdots\ 0.6080\ \ 0.3260)$
global minimum located at x =8.0241, y = 9.1465 with $f(x,y)$ =
-12.1190, search space: $x,y \in [0, 10]$ and $J_{min} \le$ -12.1189

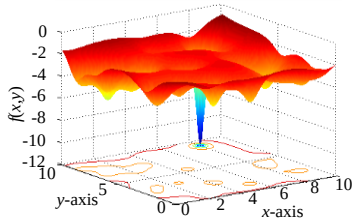


Table 2: CS and ACS Parameter Setting

CS Parameter Settings				
Test functions	n	R	N	I_{max}
BF	50	0.1	50	1,000
DJF	50	5.0	100	1,000
GF	100	0.5	100	1,000
MF	50	2.0	50	1,000
SF	100	1.5	50	1,000

ACS Parameter Settings						
Test functions	n	R	N	I_{max}	R – *adjustment*	
					Stage I	*Stage II*
BF	50	0.1	50	1,000	$J < 1 \times 10^{-2} \rightarrow R = 0.01$	$J < 1 \times 10^{-4} \rightarrow R = 0.001$
DJF	50	5.0	100	1,000	$J < 1 \times 10^{2} \rightarrow R = 1.00$	$J < 10 \rightarrow R = 0.50$
GF	100	0.5	100	1,000	$J < 1 \times 10^{-2} \rightarrow R = 0.01$	$J < 1 \times 10^{-4} \rightarrow R = 0.001$
MF	50	2.0	50	1,000	$J < 1 \rightarrow R = 0.01$	$J < 0 \rightarrow R = 0.001$
SF	100	1.5	50	1,000	$J < -2 \rightarrow R = 1.00$	$J < -5 \rightarrow R = 0.50$

Table 3: Results of Unconstrained Optimization Problems

Test functions	GA	TS	CS	ACS
BF	480,100 ± 34,612(86%)	465,100 ± 19,745(96%)	241,310 ± 9,867(100%)	200,250 ± 7,572(100%)
DJF	385,400 ± 26,748(77%)	352,520 ± 21,632(94%)	124,212 ± 8,410(98%)	112,028 ± 7,046(100%)
GF	902,150 ± 67,223(82%)	780,464 ± 41,947(92%)	514,575 ± 31,108(96%)	501,020 ± 21,663(100%)
MF	200,375 ± 22,591(88%)	145,680 ± 10,866(95%)	67,260 ± 3,103(99%)	49,640 ± 2,458(100%)
SF	685,450 ± 45,411(90%)	545,120 ± 33,012(97%)	325,850 ± 14,102(100%)	302,210 ± 9,815(100%)

Results obtained are summarized in Table 3, where the global optima are reached. The numbers are in the format: average number of evaluations (success rate). For example, 200,250±7,572(100%) means that the average number (mean) of function evaluations is 200,250 with a standard deviation of 7,572. The success rate of finding the global optima for this algorithm is 100%. Referring to Table 3, it can be observed that the ACS is much more efficient in finding the global optima with higher success rates. For all test functions, the proposed ACS outperforms GA, TS and CS.

3.2 Constrainted Optimization

For constraint optimization test, GA, TS, CS and ACS are tested against three constrained optimization problems consisting of Fcon01, Fcon02 and Fcon03, detailed as follows.

The function Fcon01 [17] is stated in equation (1). This function is a minimization problem with two design variables and two inequality constraints. The unconstrained objective function $f(x)$ has a minimum solution at (3, 2) with a corresponding function value equal to zero. The constrained minimum solution is located at $x^* = (2.246826, 2.381865)$ with an objective function value equal to $f^*(x) = 13.59085$.

$$\begin{aligned} &\min_{x} f(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2 \\ &\text{subject to,} \\ &g_1(x) = 4.84 - (x_1 - 0.05)^2 - (x_2 - 2.5)^2 \geq 0, \\ &g_2(x) = x_1^2 + (x_2 - 2.5)^2 - 4.84 \geq 0, \\ &0 \leq x_1 \leq 6, \quad 0 \leq x_2 \leq 6 \end{aligned} \tag{1}$$

For this function, parameter settings for the GA follow MATLAB-GA Toolbox [13] and for the TS follow [15]. The common search parameters of the CS are: n = 1,000, R = 0.1, N = 500 and I_{max} = 1,000. $J_{\min} \leq$ 13.60 is the minimum values of objective functions required to terminate the search. For the ACS, *R*-

adjustment is set as $J < 25.00 \rightarrow R = 0.01$ and $J < 15.00 \rightarrow R = 0.001$. Results are summarized in Table 4 and Figure 4.

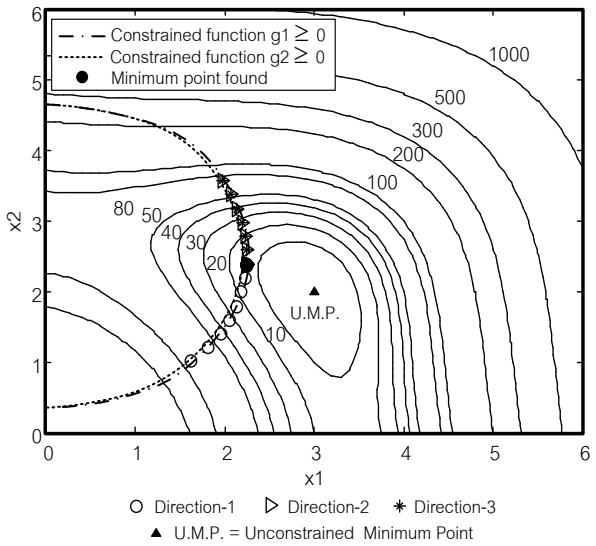


Fig. 4: Movements Of ACS Algorithm Over Fcon01

From Figure 4, it was found that the ACS spent only three search directions to reach the global optimum. Table 4 shows the best solutions found by the GA, TS, CS and ACS as well as their corresponding objective function values. It was noted that the ACS result is very close to the optimal solution and outperforms other three results not only in the aspect of the objective function values but also in that of constraint accuracy.

The function Fcon02 [18] with two design variables and two constraints is stated in equation (2). The minimum solution is located at $x^* = (0.82288, 0.91144)$ with an objective function value equal to $f^*(x) = 1.3935$.

Once GA follows MATLAB-GA Toolbox [13] and TS follows [15]. The common parameters of the CS are: $n = 1{,}000$, $R = 0.1$, $N = 500$ and $I_{max} = 1{,}000$. $J_{\min} \leq 1.3936$ is the minimum values of objective functions required to terminate the search. For the ACS, R-adjustment is set as $J < 5.00 \rightarrow R = 0.01$ and $J < 2.00 \rightarrow R = 0.001$. Results are summarized in Table 5. It was found that the ACS outperforms other three results in both the objective function values and the constraint accuracy.

Table 4: Results Of Fcon01 Constrained Optimization Problem

Algorithms	Optimal solutions x^*		Constraints		Objective function $f^*(x)$
	x_1	x_2	g_1	g_2	
GA	2.246840	2.382136	0.00002	0.22218	13.590845
TS	2.2468258	2.381863	0.00002	0.22218	13.590841
CS	2.2468262121	2.3818704379	2.21003×10^{-15}	0.2221826212	13.5908416934
ACS	2.246828512562	2.381813302772	1.8332002×10^{-18}	0.2221828329242	13.590841501204030

Table 5: Results of Fcon02 Constrained Optimization Problem

Algorithms	Optimal solutions x^*		Constraints		Objective function $f^*(x)$
	x_1	x_2	g_1	g_2	
GA	0.8350	0.9125	1.0×10^{-2}	-7.0×10^{-2}	1.3772
TS	0.8343	0.9121	5.0×10^{-3}	5.4×10^{-3}	1.3370
CS	0.82283310	0.91144165	1.0×10^{-6}	4.9×10^{-6}	1.3935867
ACS	0.82283256086	0.91142060358	-8.6463030×10^{-6}	4.9127562×10^{-5}	1.39356948924

$$\begin{aligned}
&\min_x f(x) = (x_1-2)^2 + (x_2-1)^2 \\
&\text{subject to,} \\
&g_1(x) = x_1 - 2x_2 + 1 = 0, \\
&g_2(x) = -\frac{x_1^2}{4} - x_2^2 + 1 \geq 0, \\
&-10 \leq x_1 \leq 10, \quad -10 \leq x_2 \leq 10
\end{aligned} \tag{2}$$

The function Fcon03 [17] stated in equation (3) has seven design variables and four nonlinear constraint functions. The best known optimum solution is located at $x^* = (2.330499, 1.951372, -0.4775414, 4.365726, -0.6244870, 1.038131, 1.594227)$ with the corresponding objective function equal to $f^*(x) = 680.6300573$.

$$\begin{aligned}
&\min_x f(x) = (x_1-10)^2 + 5(x_2-12)^2 + x_3^4 \\
&\qquad + 3(x_4-11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 \\
&\qquad - 4x_6x_7 - 10x_6 - 8x_7 \\
&\textit{subject to,} \\
&g_1(x) = 127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \geq 0, \\
&g_2(x) = 282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geq 0, \\
&g_3(x) = 196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \geq 0, \\
&g_4(x) = -4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geq 0, \\
&-10 \leq x_i \leq 10, \quad (i = 1, \ldots, 7)
\end{aligned} \tag{3}$$

Like Fcon01 and Fcon02, parameter settings for the GA follow MATLAB-GA Toolbox [13] and for the TS follow [15]. The common search parameters of the CS are: n = 2,000, R = 0.1, N = 1,000 and I_{max} = 1,000. $J_{\min} \leq 680.63$ is the minimum values of objective functions required to terminate the search. For the ACS, R-adjustment is set as $J < 1{,}000 \rightarrow R = 0.01$ and $J < 750 \rightarrow R = 0.001$. Results are summarized in Table 6. It was found also that the ACS outperforms other three algorithms and provides superior results.

Table 6: Results Of Fcon03 Constrained Optimization Problem

Optimal solutions x^* and Objective function $f^*(x)$	GS	TS	CS	ACS
x_1	2.32345617	2.33047	2.33087488	2.324971068354062
x_2	1.951242	1.95137	1.95136990	1.950016515825672
x_3	-0.448467	-0.47772	-0.47459349	-0.491430343898877
x_4	4.3619199	4.36574	4.36555341	4.370929879461707
x_5	-0.630075	-0.62448	-0.62452549	-0.623883321164803
x_6	1.03866	1.03794	1.03793634	1.040859835982587
x_7	1.605384	1.59414	1.59406525	1.593387864833378
$f^*(x)$	680.6413574	680.6380577	680.6350771	680.6304164174447

IV. ACS Application to ALB Problems

The application of the ACS to energy resource management of the assembly line balancing (ALB) problems is proposed. The ALB problem is considered as one of the classical industrial engineering problems [19]. An assembly line is a sequence of workstations connected together by a material handling system. It is used to assemble components or tasks into a final product. The fundamental of the line balancing problems is to assign the tasks to an ordered sequence of workstations that minimize the amount of the idle time of the line, whereas satisfying two particular constraints. The first constraint is that the total processing time assigned to each workstation should be less than or equal to the cycle time. The second is that the task assignments should follow the sequential processing order of the tasks or the precedence constraints.

The ALB can be considered as the class of combinatorial optimization problems known as NP-hard [20,21]. In this work, the single-model assembly line balancing problem is considered. Balancing of the lines can be measured by the idle time (T_{id}), the workload variance (w_v) and the line efficiency (E) [22]. Therefore, the goals of balancing assembly lines are to minimize the idle time and the workload variance. Analytical formulations for the ALB problems are stated in equations (4) - (7), where m is the number of workstations, W is the total processing time, c is the cycle time and T_i is the processing time of the i^{th} workstation.

$$m = W / c \tag{4}$$

$$T_{id} = \sum_{i=1}^{m} (c - T_i) \tag{5}$$

$$w_v = \frac{1}{m} \sum_{i=1}^{m} [T_i - (W/m)]^2 \tag{6}$$

$$E = \sum_{i=1}^{m} T_i / (m \times c) \tag{7}$$

The ALB problem can be formulated as the multiobjective constrained optimization problem. The multiobjective function (J) consisting of the workload variance (w_v) and the idle time (T_{id}) is performed as stated in equation (8), where γ_1 and γ_2 are weighted factors ($\gamma_1 + \gamma_2 = 1.0$). J in equation (8) will be minimized by the ACS metaheuristics as expressed in equation (9).

$$J = \gamma_1 \times w_v + \gamma_2 \times T_{id} \tag{8}$$

$$\begin{aligned} &\textbf{minimize} \quad J \\ &subject to \quad T_i \leq c, \\ &\qquad\qquad precedent constrain \end{aligned} \tag{9}$$

By this approach, the workload variance will be minimized in sense of the resource management optimization. This implies that, if all machines (once one machine, one workstation is assumed) of the assembly line are equally operated, resource management is optimal. This means that the assembly line of interest is balanced and all machines possess the same workload. Again by this approach, the idle time will be minimized in sense of the energy management optimization. This implies that, if all machines of the assembly line are operated with full-time production (without or with the least idle time), energy management is optimal. In contrast, running some machines without production is the energy loss of the line. In this application, the proposed ACS associated with the heuristic sequencing (HS) technique is

conducted to solve the ALB problems. The proposed ACS is used to address the number of tasks assigned for each workstation, while the HS technique is conducted to assign the sequence of tasks for each workstation according to precedence constraints as represented by Figure 5.

The HS is based on the heuristic logics of practicing engineers to arrange the sequence of tasks assigned for each workstation. In practice, assigning task will be considered from its processing time, number of succeeding tasks and number of precedent tasks. The proposed PH algorithm is thus described as follows.

Step 1 Let number of tasks be n.

Step 2 Initialize sequence of tasks $\Delta = \varnothing$ and $i = 1$.

Step 3 If a current task, δ_i, possesses properties:

(3.1) no precedent tasks,

(3.2) maximum succeeding tasks, and

(3.3) maximum processing time, put δ_i into Δ respectively, delete δ_i, $n = n - 1$, then go to Step 4, otherwise, update $i = i + 1$, then go back to Step 3.

Step 4 If $n = 0$, terminate the sequencing process. The sequence of tasks stored in Δ is successfully arranged, otherwise set $i = 1$, go back to Step (3).

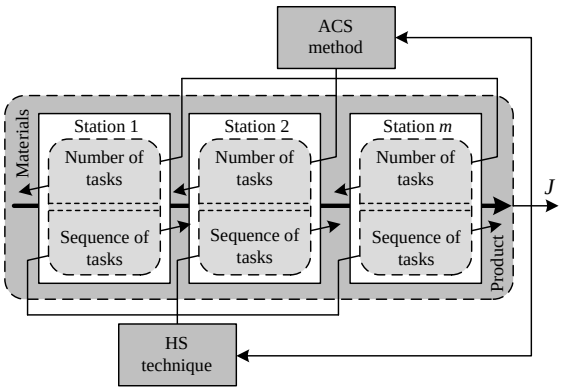


Fig. 5: ACS-Based ALB Problem Solving

The proposed approach is tested against four benchmark single-model ALB problems collected by Scholl [23] as summarized in Table 7. In this work, the number of workstations, m, can be calculate by equation (4). For all tests, the parameter settings for the GA follow MATLAB-GA Toolbox [13] and for the TS follow [15]. The common search parameters of the CS are: n = 1,000, R = 0.2 and I_{max} = 1,000. N = 500 is required to terminate the search. For the ACS, R-adjustment is set as $I_{max} = 500 \rightarrow R = 0.1$ and $I_{max} = 750 \rightarrow R = 0.01$. The CS and ACS algorithms were coded by MATLAB running on Intel Core2 Duo 2.0 GHz 3 Gbytes DDR-RAM computer. The $\gamma_1 = \gamma_2 = 0.5$ are set to compromise the w_v and T_{id} in equation (8). Table 8 provides the boundaries of number of tasks for each workstation set for the corresponding search spaces. Once the search process stopped, results obtained are summarized in Table 9.

Table 7: Details of Benchmark ALB Problems

Entry	Name	n	W(min.)	c(min.)	m
1	Buxey	29	324	50	7
2	Sawyer	30	324	40	9
3	Warnecke	58	1,548	160	10
4	Tonge	70	3,510	325	11

Table 8: Lists of Search Spaces

Entry	Search spaces
1	$S_1 \in [2, 7]$; $S_2 \in [3, 8]$; $S_3 \in [2, 6]$; $S_4 \in [3, 7]$; $S_5 \in [2, 7]$; $S_6 \in [2, 5]$; $S_7 \in [2, 5]$
2	$S_1 \in [2, 5]$; $S_2 \in [2, 6]$; $S_3 \in [3, 6]$; $S_4 \in [3, 6]$; $S_5 \in [2, 5]$; $S_6 \in [1, 4]$; $S_7 \in [1, 5]$; $S_8 \in [1, 4]$; $S_9 \in [2, 5]$
3	$S_1 \in [3, 6]$; $S_2 \in [3, 6]$; $S_3 \in [4, 8]$; $S_4 \in [6, 9]$; $S_5 \in [5, 9]$; $S_6 \in [3, 6]$; $S_7 \in [4, 7]$; $S_8 \in [5, 8]$; $S_9 \in [5, 8]$; $S_{10} \in [4, 8]$
4	$S_1 \in [6, 9]$; $S_2 \in [5, 8]$; $S_3 \in [2, 6]$; $S_4 \in [3, 6]$; $S_5 \in [5, 7]$; $S_6 \in [3, 6]$; $S_7 \in [5, 10]$; $S_8 \in [6, 12]$; $S_9 \in [6, 12]$; $S_{10} \in [5, 8]$; $S_{11} \in [5, 8]$;

Note: S_i stands for the i^{th} workstation.

Table 9: Results of ALB Problems

Entry 1: Buxey			
Apps.	w_v	T_{id}(min.)	E(%)
GA	11.06	26.00	94.46
TS	8.77	26.00	95.02
CS	2.64	26.00	95.38
ACS	1.35	26.00	96.42
Entry 2: Sawyer			
Apps.	w_v	T_{id}(min.)	E(%)
GA	6.00	36.00	90.00
TS	5.84	36.00	91.85
CS	3.22	36.00	95.47
ACS	1.56	36.00	97.30
Entry 3: Warnecke			
Apps.	w_v	T_{id}(min.)	E(%)
GA	18.56	52.00	95.66
TS	10.11	52.00	95.91
CS	8.86	52.00	96.04
ACS	6.56	52.00	97.35
Entry 4: Tonge			
Apps.	w_v	T_{id}(min.)	E(%)
GA	11.54	65.00	98.18
TS	8.74	65.00	98.85
CS	3.98	65.00	99.01
ACS	1.17	65.00	99.40

Note: Apps. stands for approaches.

Table 10: Results of Tonge-ALB Problems by ACS

Work-Station (m)	Tasks assigned for each workstation	Station Time (min.)	Idle Time (min.)
1	1,2,3,5,9,15,24	317	8
2	4,6,7,10,41,68	319	6
3	11,16,18	318	7
4	8,12,14,30,69	318	7
5	13,17,19,20,22,57	319	6
6	21,23,25,27	320	5
7	26,29,31,32,33,34,58,59	320	5
8	28,35,36,37,38,44,45,48,51,70	319	6
9	39,40,56,61,62,63,64	320	5
10	42,43,46,47,49,52,67	321	4
11	50,53,54,55,60,65,66	319	6

The total idle time (T_{id}) = 65.00 min.,
The workload variance (w_v) = 1.17,
The line efficiency (E) = 99.40%.

Referring to Table 9, it was found that the proposed ACS associated with the HS technique is capable of producing solutions superior to other techniques for all ALB problems. The workload variance, w_v, can be successfully minimized in sense of the resource management optimization, while the idle time, T_{id}, can be satisfactory minimized in sense of the energy management optimization. For example, Table 10 contains the results of the Tonge-ALB problem obtained by the ACS with HS association in details.

V. Conclusions

The application of the adaptive current search (ACS) to energy resource management optimization has been proposed in this paper. The ACS is the modified version of the current search (CS) developed from the behavior of an electric current flowing through electric networks. With the memory list (ML) and the adaptive radius (AR) mechanism, the ACS effectiveness can be improved to escaping from any local entrapment and to speed up the search process. For its performance evaluation against five benchmark unconstrained and three constrained optimization problems compared with genetic algorithm (GA), tabu search (TS) and CS, it was found that the ACS outperforms other algorithms and provides superior results which are very close to the optimal solutions for all testes functions. The proposed ACS has been applied to energy resource management of the assembly line balancing (ALB) problems associated with the heuristic sequencing (HS) technique. By this approach, the ACS is applied to address the number of tasks for each station, while the HS technique is used to assign the sequence of tasks according to precedence constraints. The workload variance (w_v) and the idle time (T_{id}) are formed as the multipleobjective function for this application. As results from four benchmark ALB problems compared with the GA, TS and CS, it was found that the ACS associated with the HS technique is capable of producing solutions superior to other techniques for all ALB problems. The workload variance can be successfully minimized in sense of the resource management optimization, while the idle time can be satisfactory minimized in sense of the energy management optimization. This can be concluded that the proposed ACS is an alternative powerful algorithm to solve the ALB problems in sense of energy resource management and other optimization problems.

References

[1] W. C. Turner. Energy Management Handbook Fairmont Press, USA., 2004.

[2] B. L. Capehart, W. C. Turner, W. J. Kennedy. Guide to Energy Management. Fairmont Press, USA., 1983.

[3] S. S. Rao. Engineering Optimization: Theory and Practice. John Wiley & Sons, 2009.

[4] X. S. Yang. Engineering Optimization: An Introduction with Metaheuristic Applications. John Wiley & Sons, 2010.

[5] D. T. Pham, D. Karaboga. Intelligent Optimization Techniques. Springer, London, 2000.

[6] F. Glover, G. A. Kochenberger. Handbook of Metaheuristics. Kluwer Academic Publishers, 2003.

[7] E. G. Talbi. Metaheuristics form Design to Implementation. John Wiley & Sons, 2009.

[8] X. S. Yang. Nature-Inspired Metaheuristic Algorithms. Luniver Press, 2010.

[9] A. Sukulin, D. Puangdownreong. A novel meta-heuristic optimization algorithm: current search. Proceeding of the 11th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases (AIKED '12), 2012, pp.125-130.

[10] A. Sukulin, D. Puangdownreong. Control synthesis for unstable systems via current search. Proceeding of the 11th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases (AIKED '12), 2012, pp.131-136.

[11] A. Sukulin, D. Puangdownreong. Current search and applications in analog filter design problems. Communication and Computer, v9, n9, 2012, pp. 1083-1096.

[12] D. E. Goldberg. Genetic Algorithms in Search Optimization and Machine Learning. Addison Wesley Publishers, Boston, 1989.

[13] MathWorks. Genetic Algorithm and Direct Search Tool- box: For Use with MATLAB. User's Guide, v1, MathWorks, Natick, Mass, 2005.

[14] F. Glover. Tabu search - part I. ORSA Journal on Computing, v1, n3, 1989, pp.190-206.

[15] F. Glover. Tabu search - part II. ORSA Journal on Computing, v2, n1, 1990, pp.4-32.

[16] M. M. Ali, C. Khompatraporn, Z. B. Zabinsky. A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. Journal of Global Optimization, v31, n4, 2005, pp.635-672.

[17] K. Deb. An efficient constraint handing method for genetic algorithms. Computer Methods in Applied Mechanics and Engineering, v186, 2000, pp. 311-338.

[18] J. Braken, G. P. McCormick. Selected Applications of Nonlinear Programming. John Wiley & Sons, New York, 1968.

[19] A. L. Gutjahr, G. L. Nemhauser. An algorithm for the balancing problem. Management Science, v11, 1964, pp.23-25.

[20] M. D. Kilbridge, L. Wester. A heuristic method of assembly line balancing. The Journal of Industrial Engineering, v12, n4, 1961, pp.292-298.

[21] A. L. Arcus. COMSOAL: a computer method of sequencing operations for assembly line. International Journal of Production Research, v4, 1966, pp.25-32.

[22] M. Amen. Heuristic methods for cost-oriented assembly line balancing: a comparison on solution quality and computing time. International Journal of Production Economics, v69, 2001, pp.255-264.

[23] A. Scholl. Balancing and Sequencing of Assembly Lines. Physica, Heidelberg, 1999.

Authors' Profiles

Supaporn Suwannarongsri (1983 —), female, Bangkok, Thailand, Assistant Professor, Ph.D. candidate, she received the B.Eng. degree in Industrial Engineering from Thonburi University (TRU), Bangkok, Thailand, in 2004 and the M.Eng. degree in Industrial Engineering from King Mongkut's Institute of Technology Ladkrabang (KMITL), Bangkok, Thailand, in 2008, respectively.

Since 2006, she has been with the Department of Industrial Engineering, Faculty of Engineering, South-East Asia University (SAU), Bangkok, Thailand, where she is currently an Assistant Professor of Industrial Engineering. She has published over 40 refereed journal and conference papers in the areas of operation research, production planning and mehaheuristic optimization. She is now the Ph.D. candidate and pursuing her Ph.D. degree in Sustainable Energy and Environment at Rattanakosin College for Sustainable Energy and Environment (RCSEE), Rajamangala University of Technology Rattanakosin, Nakhon Pathom, Thailand. Her research interests include operation research, production planning and design, mehaheuristic optimization, and applications of mehaheuristic algorithms to various real-world industrial engineering problems.

Tika Bunnag (1967 —), male, Bangkok, Thailand, supervisor for Ph.D. candidate, he received the B.Eng. (Mechanical), M. Eng. (Thermal Technology) and Ph.D. (Energy Technology) from King Mongkut's University of Technology Thonburi, in 1993, 1995 and 2003 respectively.

In 2011, he joined the Rattanakosin College for Sustainable Energy and Environment, Rajamangala University of Technology Rattanakosin, Thailand, as a Vice Director and Head of Technology Management Research center. His representative published articles lists as follow: Experimental study of a Roof Solar Collector towards the Natural Ventilation of New Habitations (World Renewable Energy Congress, 1996), Experimental Investigation of Free Convection in an Open Ended Vertical Rectangular Channel (Comples journal, 2002), Experimental Investigation of Free Convection in Open Ended Inclined Rectangular Channel Heated from the Top (The Internal Journal of Ambient Energy, 2003), Improvement of Thermal Performance of Double Glass Skylight in Thailand (Asia-Pacific Regional Conference of International Solar Energy Society, 2004) and The Reduce of Building Cooling Load from Double Glass Skylight in Bangkok (Sustainable Energy and Green Architecture III, 2011) etc.

His research interests include Advance Energy Management, Magnetic Free Energy Generator, Green Buildings, Zero Waste Management and Sustainable Energy Navigation.

Waraporn Klinbun (1983 —), female, Bangkok, Thailand, supervisor for Ph.D. candidate, she received a bachelor degree in Biotechnology, Master and Ph.D. degree in Mechanical Engineering from Thammasat University, Thailand. Her research focuses on the electromagnetic energy analysis in various fields,

including heating and drying process, microwave technology, transportation phenomena by using computational fluid dynamic method such as finite volume. Her works are well received internationally and have been presented and published in the first and second tier conferences and journals.

Research areas that she currently focuses on are the analysis of mathematical modeling of various processes related to electromagnetic heating of dielectric materials, transportation in porous media, including bioengineering process.

How to cite this paper: Supaporn Suwannarongsri, Tika Bunnag, Waraporn Klinbun,"Energy Resource Management of Assembly Line Balancing Problem using Modified Current Search Method", International Journal of Intelligent Systems and Applications(IJISA), vol.6, no.3, pp.1-11, 2014. DOI: 10.5815/ijisa.2014.03.01

