Task Assignment Problem Solved by Continuous Hopfield Network

ETTAOUIL Mohamed[1], LOQMAN Chakir[2], HAMI Youssef[3] and HADDOUCH Khalid[4]

**[1,3,4] UFR: Scientific Computing and Computer sciences, Engineering sciences,
Modeling and Scientific Computing Laboratory, Faculty of Science and Technology.
University Sidi Mohamed Ben Abdellah, Box 2202, Fez, Morocco**

**[2] Department of Computer Engineering, High School of technology,
Moulay Ismail University, B. P. 3103, 50000, Toulal, Meknes, Morocco**

Abstract

The task assignment problem with non uniform communication costs (TAP) consists in finding an assignment of the tasks to the processors such that the total execution and communication costs is minimized. This problem is naturally formulated as 0-1 quadratic programming subject to linear constraints (QP). In this paper, we propose a new approach to solve the task assignment problem with non uniform communication costs using the continuous Hopfield network (CHN). This approach is based on some energy or Lyapunov function, which diminishes as the system develops until a local minimum value is obtained. We show that this approach is able to determine a good solution for this problem. Finally, some computational experiments solving the task assignment problem with non-uniform communication costs are shown.

Keywords: *Combinatorial optimization, Continuous Hopfield network, Multiprocessor systems, Quadratic 0-1 programming, Task assignment problem.*

1. Introduction

The task assignment problem play a vital role in a computation system with a number of distributed processors, where a set of tasks must be assigned to a set of processors minimizing the sum of execution costs and communication costs between tasks. This problem has been proved to be an NP-hard problem [26]. Several variants of the task allocation problem have been considered, with different architecture of distributed system or in structure of costs [1]- [2]-[3]-[13]-[21]. They are basically divided into three categories:

• The graph theoretical minimizes the total interprocessor communication cost by performing a partitioning algorithm on the graph such that, each partition includes a set of tasks, which are assigned to specified single processor [22].

• Integer programming using column generation or branch-and-bound techniques can be used to solve the problem more efficiently [5]-[6].

• Meta-heuristic involving genetic algorithm [18]-[25] and simulated annealing [19]-[15] have been used to derive approximate solutions with reasonable time [14] they are applicable to larger dimensional problems.

The task assignment problem with non-uniform communication costs can be modeled as 0-1 quadratic programming which consists in minimizing a quadratic function subject to linear constraints (QP). To solve the QP problem, many different methods are tried and tested such as interior point, semi definite relaxations and lagrangian relaxations [8]-[9]. In this paper, we introduce a new approach using the continuous Hopfield network for solving the QP problem.

Hopfield neural network was introduced by Hopfield and Tank [16]-[18]. It was first applied to solve combinatorial optimization problems. It has been extensively studied, developed and has found many applications in many areas, such as pattern recognition, model identification, and optimization. It has also demonstrated capability of finding solutions to difficult optimization problems [12]. The interesting steps for this method are to define the generalized energy function for solving any combinatorial problem and to determine the parameters setting of CHN [24]-[11].

In this paper, our main objective is to propose a new method to solve the TAP using the continuous Hopfield network. This paper is organized as follows. In section 2, we provide a formulation of Task Assignment Problem with non uniform communication cost as a 0-1 quadratic programming (QP). In section 3, we describe the most interesting steps for solving the QP problem using the

continuous Hopfield network. The experimental results are presented in section 4.

2. Problem Formulation

The task assignment problem with non uniform communication costs consists in finding an assignment of N tasks to M processors such that the total execution and communication costs is minimized. This problem is stated as a two sets and two parameters where:

- $T = \{T_1,...,T_N\}$ a set of N tasks.
- $P = \{P_1,...,P_M\}$ a set of M processors.
- The execution cost e_{ik} of task i if is assigned to processor k.
- The communication cost c_{ikjl} between two different tasks i and j if they are respectively assigned to processors k and l.

In the following, we want to present a formulation of the task assignment problem as a 0-1 quadratic programming [7].

For each task $i \in \{1,..,N\}$, we introduce M binary variables x_{ik}, $k \in \{1,..,M\}$, such that:

$$x_{ik} = \begin{cases} 1 & \text{If the task } i \text{ is assigned to processor } k \\ 0 & \text{Otherwise} \end{cases}$$

This matrix is converted to a n-vector:

$x = (x_{11}, x_{12},..., x_{1M}, x_{21}, x_{22},..., x_{2M},..., x_{N1}, x_{N2},..., x_{NM})^T$ with $n = N \times M$

- Each task should be assigned to exactly one processor:

$$\sum_{k=1}^{M} x_{ik} = 1 \qquad \forall\, i \in \{1,...,N\}$$

The linear constraints can be rewritten as :

$$\sum_{k=1}^{M} x_{ik} = 1,\ i = 1,...,N \quad \Leftrightarrow \quad Ax = b$$

The matrix $A \in IR^{N \times n}$ with $n = N \times M$ and the vector $b \in IR^N$ of the linear constraints are:

$$A = \begin{pmatrix} 1 & \cdots & \cdots & 1 & \cdots & \cdots & 0 & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & 0 & \cdots & \cdots & \vdots & & & \vdots \\ \vdots & & & \vdots & & & 0 & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & 0 & \cdots & \cdots & 1 & \cdots & \cdots & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

The main objective is to minimize the total execution and communication costs incurred by the task assignment subject to resources constraint. Then, we can define the objective function $F(x)$ in the following way:

$$F(x) = \sum_{i=1}^{N-1} \sum_{k=1}^{M} \sum_{j=i+1}^{N} \sum_{l=1}^{M} c_{ikjl}\, x_{ik}\, x_{jl} + \sum_{i=1}^{N} \sum_{k=1}^{M} e_{ik}\, x_{ik}$$
$$= x^t C x + e^t x$$

Where C is $n \times n$ matrix with the general term is denoted by c_{ikjl}.

The first and second terms in the objective function represent the total execution cost and communication cost, respectively, incurred by the assignment $x = (x_{11},..., x_{1M},..., x_{N1},..., x_{NM})$.

Finally, we obtain the following 0-1 quadratic program (QP) with a quadratic function subject to linear constraints representing the TAP problem with n variables and N linear constraints:

$$(QP) \quad \begin{cases} Min & f(x) = x^t C x + e^t x \\ Subject\ to & \\ & Ax = b \\ & x \in \{0,1\}^n \end{cases}$$

Therefore, C is $n \times n$ matrix with the general term is denoted by c_{ikjl}, A is $N \times n$ matrix and $b \in IR^N$. Without lost of generality, we can suppose that C is symmetric and also that diagonal terms of C are equal to 0. If this matrix is not symmetric, it can be converted to the symmetric form $\frac{C + C^t}{2}$ and the linear terms $c_{ikik} x_{ik}$ can be substituted for the diagonal terms $c_{ikik} x_{ik}^2$, because $x_{ik}^2 = x_{ik}$ for $x_{ik} \in \{0,1\}$.

Then, we obtain the following QP problem:

$$(QP) \quad \begin{cases} Min & f(x) = \frac{1}{2} x^t Q x + e^t x \\ Subject\ to & \\ & Ax = b \\ & x \in \{0,1\}^n \end{cases}$$

where $Q = C + C^t$.

Although the QP problem is NP-hard [4]-[26], some special instances are polynomial time solvable. Those instances are solvable in $O(NM^2)$ time in heterogeneous

networks [4].
In this work, our objective is to solve the task assignment problems using the continuous Hopfield networks. Then, in this case, the most important step consists of representing or mapping the TAP in the form of the energy function associated with the continuous Hopfield networks. According to the QP model, we define the associated energy function and the parameters setting. Therefore, the continuous Hopfield networks can be used to solve the task assignment problem.

3. Task Assignment Problem solved by CHN

Hopfield neural network was introduced by Hopfield and Tank [16]-[17]. It was first applied to solve combinatorial optimization problems. As can be noticed, after modeling the task assignment problem into a 0-1 quadratic programming with a quadratic function subject to linear constraints, we present a general method for solving the TAP problems using the continuous Hopfield networks.

3.1 Using the continuous Hopfield network to solve QP problem

The continuous Hopfield neural network is a fully connected neural network i.e. the n neurons of the CHN are fully connected, which means that every neuron is connected to all other neurons. Let T_{ij} be the strength of the connection from neuron j to neuron i. Each neuron i has an offset bias i_i^b. The current state and the output of the neuron i are respectively represented by u_i and x_i [23].
The dynamics of the *CHN* are described by the differential equation:

$$\frac{du}{dt} = -\frac{u}{\tau} + Tx + i^b \qquad (1)$$

Where u, x and i^b are the vectors of neuron states, outputs and biases. The output function $x_i = g(u_i)$ is a hyperbolic tangent, which is bounded below by 0 and above by 1.

$$g(u_i) = \frac{1}{2}(1+\tanh(\frac{u_i}{u_0})) \quad \text{where} \quad u_0 > 0 \quad \text{and} \quad i = 1,...,n$$

Where u_0 is a parameter used to control the gain of the activation function. In order to use the continuos Hopfiled network, for solving any combinatorial problems, we should be reformuled this latter into energy function associated to the CHN. This energy function is defined by the following expression [23]:

$$E(x) = -\frac{1}{2}x^t Tx - (i^b)^t x. \qquad (2)$$

Typically, in the CHN, the energy function is made equivalent to the objective function which is to be minimized, while each of the constraints of the optimization problem are included in the energy function as penalty terms.

3.2 Energy function and parameter-setting for the TAP problem

In order to solve the task assignment problem using the continuous Hopfield networks, we define the generalized energy function for the TAP problems basing on the model. Recall that, the task assignment problem are modeled as 0-1 quadratic programming with n variables and N linear constraints.

$$(QP) \quad \begin{cases} Min & f(x) = \frac{1}{2}x^t Qx + e^t x \\ Subject\ to & \\ & Ax = b \\ & x \in \{0,1\}^n \end{cases}$$

The generalized energy function allows representing mathematical programming problems with quadratic objective function and linear constraints. This energy function includes the objective function $f(x)$ and it penalizes the linear constraints $Ax = b$ with a quadratic term and a linear term.
The generalized energy function for the QP problem is defined by [24]:

$$E(x) = \frac{\alpha}{2}x^t Qx + \alpha e^t x + \frac{1}{2}(Ax)^t \Phi (Ax) + x^t diag(\gamma)(1-x) + \beta^t Ax \qquad (3)$$

With $x \in [0,1]^n$, $\alpha \in IR^+$, $\beta \in IR^N$, $\gamma \in IR^n$ and Φ is an $N \times N$ symmetric matrix. Here $diag(\gamma)$ denotes the diagonal matrix constructed from the vector γ.
To define the energy function of the (QP) problem, the following considerations need to be taken into account so that the mathematical expression of energy function (3) is simplified.

• Only the main diagonal terms of the quadratic matrix parameter Φ are considered:

$$\Phi_{kj} = \begin{cases} 0 & if\ j \neq k \\ \phi & if\ j = k \end{cases}$$

Where ϕ is a positif scalar.

• All linear constraints are equally weighted, where β is the associated parameter.
• The parameter penalizing the non-extreme values of x_{ik} is γ.

Consequently, the following generalized energy function of the QP problem is proposed:

$$E(x) = \frac{\alpha}{2} x^t Q x + \alpha e^t x + \frac{1}{2}\phi (Ax)^t (Ax) + x^t diag(\gamma)(1-x) + \beta^t Ax \quad (4)$$

So, the following generalized energy function in algebraic form of QP problem must also be defined by:

$$E(x) = \frac{\alpha}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{j=1}^{N}\sum_{l=1}^{M} c_{ijkl}\, x_{ik}\, x_{jl} + \alpha\sum_{i=1}^{N}\sum_{k=1}^{M} e_{ik}\, x_{ik} + \frac{1}{2}\phi\sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{l=1}^{M} x_{ik}\, x_{il} + \beta\sum_{i=1}^{N}\sum_{k=1}^{M} x_{ik} + \gamma\sum_{i=1}^{N}\sum_{k=1}^{M} x_{ik}(1-x_{ik}) \quad (5)$$

To determine the weights and thresholds, we use the assimilation between equation (2) and the algebraic form of the generalized energy function. Then, the weights and thresholds of the connections between n neurons are:

$$\begin{cases} T_{ikjl} = -\alpha c_{ikjl} - \delta_{ij}\phi + 2\delta_{ij}\delta_{kl}\gamma \\ i_{ik}^{b} = -\alpha e_{ik} - \beta - \gamma \end{cases} \quad (6)$$

Where δ_{ij} is the Kroenecker delta such that:

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

In this way, the quadratic programming has been presented as an energy function of continuous Hopfield network. To solve an instance of the QP problem, the parameter setting procedure is used. This procedure assigns the particular values for all parameters of the network, so that any equilibrium points are associated with a valid affectation of all variables when all constraints are satisfied.
The weights and thresholds of the quadratic program depend on the parameters α, ϕ, β and γ. The parameter-setting procedure is based on the partial derivatives of the generalized energy function:

$$\frac{\partial E(x)}{\partial x_{ik}} = E_{ik}(x) = \alpha\sum_{j=1}^{N}\sum_{l=1}^{M} c_{ikjl}\, x_{jl} + \alpha e_{ik} + \phi\sum_{l=1}^{M} x_{il} + \beta + \gamma(1-2x_{ik}) \quad (7)$$

Thus, there is no communication in the same task on two processors:

$$c_{ikil} = 0 \quad \forall (i,k,l) \in \{1,..,N\}\times\{1,..,M\}^2$$

Then the derivative of the energy function becomes:

$$E_{ik}(x) = \alpha\sum_{j=1, j\neq i}^{N}\sum_{l=1}^{M} c_{ikjl}\, x_{jl} + \alpha e_{ik} + \phi\sum_{l=1}^{M} x_{il} + \beta + \gamma(1-2x_{ik}) \quad (8)$$

To solve the QP problem, the following sets are needed:
• H is a set of the Hamming hypercube :
$H = \{x \in [0,1]^n\}$
• H_C is a set of the Hamming hypercube corners :
$H_C = \{x \in H : x_i \in \{0,1\}, \forall\ i = 1,...,n\}$
• H_F is a set of feasible solutions :
$H_F = \{x \in H_C : Ax = b\}$.

This procedure uses the hyperplane method, so that the Hamming hypercube H is divided by a hyperplane containing all feasible solutions. Based on this hyperplane and the associated half-spaces [24]. The hyperplane method is briefly explained below. In order to guarantee the instability of the interior points $x \in H - H_C$, some initial conditions are imposed on some parameters:

– $T_{ikik} = -\phi + 2\gamma \geq 0$

Where $i \in \{1,..., N\}$ and $k \in \{1,..., M\}$.
The QP problem has only one family of linear constraints:

$$d_i(x) = \sum_{k=1}^{M} x_{ik} = 1 \quad \forall\, i \in \{1,..., N\}$$

The partition of the set $H_C - H_F$ is defined as :

$$H_C - H_F = W_{1,1} \cup W_{1,2}$$

– $W_{1,1} = \{\exists i : d_i(x) > 1\} \cap \{d_0(x) \geq N\}$

Where $d_0(x) = \sum_{i=1}^{N} d_i(x)$.

In this case one task has been excuted by two different processors so that $x_{ik} = x_{il} = 1$, i.e., the task i is going to excute in two different processors, which is illogical. To avoid this, the following condition should be imposed:

$$E_{ik}(x) \geq \alpha d_{\min} + 2\phi + \beta - \gamma \geq \varepsilon \qquad (9)$$

Where $d_{\min} = M(N-1)C_{min} + e_{min}$

with

$C_{min} = Min\left\{ c_{ikjl} \ / (i,j) \in \{1,...,N\}^2 \text{ and } (k,l) \in \{1,...,M\}^2 \right\}$

$e_{min} = Min\left\{ e_{ik} \ / i \in \{1,...,N\} \text{ and } k \in \{1,...,M\} \right\}$

– $W_{1,2} = \{\exists i : d_i(x) < 1\} \cap \{d_0(x) < N\}$

In this case one task has not been assigned to any processor, such that $x_{ik} = 0 \quad \forall k \in \{1,...,M\}$,i.e., any processor that not reserved to excute the task i, which is contradictory. Therefore, the following condition should be imposed:

$$E_{ik}(x) \leq \alpha d_{\max} + \beta + \gamma \leq -\varepsilon \qquad (10)$$

Where $d_{\max} = M(N-1)C_{max} + e_{max}$

with

$C_{max} = Max\left\{ c_{ikjl} \ / (i,j) \in \{1,...,N\}^2 \text{ and } (k,l) \in \{1,...,M\}^2 \right\}$

$e_{max} = Max\left\{ e_{ik} \ / i \in \{1,...,N\} \text{ and } k \in \{1,...,M\} \right\}$

Consequently, we can determine the parameters setting by resolving the following system:

$$\begin{cases} \alpha > 0 \\ \phi \geq 0 \\ -\phi + 2\gamma \geq 0 \\ \alpha d_{\min} + 2\varphi + \beta - \gamma = \varepsilon \\ \alpha d_{\max} + \beta + \gamma = -\varepsilon \end{cases} \qquad (11)$$

These parameters setting are determinate by fixing α , ε and compute the rest of parameters γ, β and ϕ :

- $\gamma = (\alpha(d_{max} - d_{min}) + 2\varepsilon)/2$
- $\beta = -\varepsilon - \alpha d_{max} - \gamma$
- $\phi = 2\gamma$.

Finally, the weights and thresholds of *CHN* as the following:

$$\begin{cases} T_{ikjl} = -\alpha c_{ikjl} + \delta_{ij}(\delta_{kl} - 1)(\alpha(d_{\max} - d_{\min}) + 2\varepsilon) \\ i_{ik}^{b} = \alpha(d_{\max} - e_{ik}) + \varepsilon \end{cases}$$

Where δ_{ij} is the Kroenecker delta.

Finally, we obtain an equilibrium point for the CHN using the algorithm described in [23], so compute the solution of task assignment problem.

4. Computational experiments

For evaluating and showing the practical interest of our approach, we have used the instances provided in [7], where coefficients e_{ik} and c_{ikjl} are randomly generated in the interval $[-50,50]$. These experiments are effectuated in personal computer with processor Intel Core i3 2,53 GHz, and 3 Go of RAM. The performance has been measured in terms of the CPU time per second. This solver is implemented by java language. In this experimentation, some instances used as [7]. The starting points are generated randomly.

$$x_{ik} = 0.99 + 10^{-3} u$$

Where $i = 1,...,N, k = 1,...,M$ and u is a random uniform variable in the interval $[-0.5,0.5]$.

The value of each parameters is determinated by solving the system (11) where $\varepsilon = 10^{-4}$ and $\alpha = \frac{1}{N}$, with N is the number of tasks.

Table (1) summarizes the results of the executions of our approach on these instances. For each size of instances, we run the algorithm 200 times and the quality of the solution obtained by our approach was evaluated by the following performance expression:

$$\rho = \frac{Optimal \quad value}{Objective \quad value \quad obtained \quad by \quad CHN}$$

Finally, the interesting results are obtained by this approach. The resolution times obtained by our approach are reasonable. From a theoretical point of view, our approach is very powerful. It can happen to solve a TAP of large size.

5. Conclusions

In this paper, we have introduced a new method to find a solution of the task assignment problem. This problem has been presented as 0-1 quadratic problem subject to linear constraints (QP). To solve this problem, we have used the continuous Hopfield network. Some numerical examples assess the effectiveness of the theoretical results are shown in this paper, and also the advantages of this new approach. Several directions can be investigated to try to improve this method, such as reducing the architecture of Hopfield neural network [10], and methods of fixation and evaluation.

Table 1: The typical instances of TAP solved by CHN

Instances	CPLEX			CHN					
	N	M	Optimal Value	best objective value	ρ			Mean iterations	Mean time (ms)
					minimum	mean	mode		
Tassnu_10_3_1	10	3	-719	-719	1	1,18	1,16	75,87	5,14
Tassnu_10_3_2	10	3	-790	-773	1,02	1,32	1,17	91,89	2,70
Tassnu_10_3_3	10	3	-624	-614	1,02	1,69	1,97	73,36	2,72
Tassnu_10_3_4	10	3	-734	-649	1,13	1,51	1,29	90,57	2,62
Tassnu_10_3_5	10	3	-871	-832	1,05	1,33	1,24	93,36	2,97
Tassnu_10_3_6	10	3	-677	-609	1,11	2,31	2,25	63,68	1,92
Tassnu_10_3_7	10	3	-613	-613	1	1,4	1,47	87,13	2,10
Tassnu_10_3_8	10	3	-495	-479	1,03	5,11	5,56	66,99	1,57
Tassnu_10_3_9	10	3	-750	-730	1,03	1,16	1,03	82,01	1,73
Tassnu_10_3_10	10	3	-486	-452	1,08	2,89	4,3	85,15	2,42
Tassnu_15_5_1	15	5	-1985	-1908	1,04	1,91	1,77	104,46	12,34
Tassnu_15_5_2	15	5	-1568	-1379	1,14	2,31	2,15	109,1	8,38
Tassnu_15_5_3	15	5	-1892	-1748	1,08	1,85	1,94	106,67	8,74
Tassnu_15_5_4	15	5	-1806	-1614	1,12	2,45	2,37	106,9	10,86
Tassnu_15_5_5	15	5	-1881	-1737	1,08	1,65	1,6	105,06	6,69
Tassnu_15_5_6	15	5	-1950	-1807	1,08	1,88	1,75	105,27	6,67
Tassnu_15_5_7	15	5	-1893	-1801	1,05	1,99	2,02	107,03	6,58
Tassnu_15_5_8	15	5	-1733	-1733	1	1,91	1,68	107	6,54
Tassnu_15_5_9	15	5	-1798	-1548	1,16	2,12	1,89	105,5	6,56
Tassnu_15_5_10	15	5	-1763	-1560	1,13	1,92	1,91	105,89	6,43

References

[1] R. K. Arora and S. P. Rana, "Analysis of the module assignment problem in distributed computing systems with limited storage", Information Processing Letters, Vol. 10, No. 3, 1980, pp. 111–115.

[2] A. Billionnet, M. C. Costa, and A. Sutter, " An efficient algorithm for a task allocation problem", Journal of the ACM, Vol. 39, No. 3, 1992, pp. 502–518.

[3] A. Billionnet and S. Elloumi, "Placement de tâches dans un système distribué et dualité lagrangienne", Revue d'Automatique, d'Informatique et de Recherche Opérationnelle (R.A.I.R.O.), série verte, Vol. 26, No. 1, 1992, pp. 83–97.

[4] S. H. Bokhari, "A shortest tree algorithm for optimal assignments across space and time in distributed processor system", IEEE T. Software Eng, vol. 7, 1981, pp. 583-589.

[5] W. W. Chu, " Optimal file allocation in multiple computin system", IEEE Trans. Comput, Vol. C-18, 1969, pp. 885-889.

[6] O. I. El-Dessouki and W. H. Huan, "Distributed ennumération on network computers", IEEE Trans. Comput, Vol. C-29, 1980, pp. 1068-1079.

[7] S. Elloumi, "The task assignment problem, a library of instances", http://cedric.cnam.fr/oc/TAP/TAP.html, 2004.

[8] M. Ettaouil and C. Loqman, "A New Optimization Model for Solving the Constraint Satisfaction Problem", Journal of Advanced Research in Computer Science, Vol. 1, 2009, pp. 13-31.

[9] M. Ettaouil and C. Loqman, "Constraint Satisfaction Problems Solved by Semidefinite Relaxations", WSEAS TRANSACTIONS on COMPUTERS, Vol. 7, 2008, pp. 951--961.

[10] M. Ettaouill and Y. Ghanou, "Neural architectures optimization and genetic algorithms", WSEAS transactions on computers, Vol. 8, No. 3, 2009, pp. 526-537.

[11] M. Ettaouill, C. Loqman and K. haddouch, "Job Shop Scheduling Problem Solved by the Continuous Hopfield Networks", Vol. 2, No. 1, 2010, pp. 31 – 47.

[12] D. J. Evansi and M. N. Sulaiman, "Solving optimisation problems using neucomp-a neural network compiler", International Journal of Computer Mathematics, Vol. 62, No. 1, 1996, pp. 1-21.

[13] W. Fernandez de la Vega and M. Lamari, "The task allocation problem with constant communication", Discrete Applied Mathematics, Vol 131, No. 1, 2003, pp. 169-177.

[14] V. B. Gylys and J. A. Edward, "Optimal partitioning of workload for distributed systems", In Dig. COMPCON, Fall 1976, pp. 353-357.

[15] Y. Hamam and K. S. Hindi, "Assignment of program tasks to processors: a simulated annealing approach", European Journal of Operational Research, Vol. 122, 2000, pp. 509–513.
[16] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities", Proceedings of the National Academy of Sciences of the United States of America, vol. 79, 1982, pp. 2554-2558.
[17] J. J. Hopfield and D. W. Tank, "Neural computation of decisions in optimization problems", Biological Cybernetics, Vol. 52, 1985, pp. 1-25.
[18] E. S. H. Hou, N. Ansari and H. Ren, "A genetic algorithm for multiprocessor scheduling", IEEE Transactions on Parallel and Distributed Systems, Vol. 5, 1994, pp. 113– 120.
[19] F. T. Lin and C. C. Hsu, "Task assignment scheduling by simulated annealing", Proceeding of Conference on Computer and Communication Systems, 1990, pp. 279– 283.
[20] S. Maw-Sheng Chern, G. H. Chen, and Pangfeng Liu, "An LC branch-and-bound algorithm for the module assignment problem", Information Processing Letters, Vol. 32, No. 2, 1989, pp. 61–71.
[21] F. Roupin, "On approximating the memory-Constrained Module Allocation Problem", Information Processing Letters, Vol. 61, No. 4, 1997, pp. 205–208.
[22] H. S. Stone, "Multiprocessor scheduling with the aid of network flow for algorithms", IEEE Trans. Software Engrg, SE, Vol. 3, No. 1, 1977, pp. 85-93.
[23] P. M. Talaván and J. Yáñez, "A continuous Hopfield network equilibrium points algorithm", Computers and Operations Research, Vol. 32, 2005, pp. 2179-2196.
[24] P. M. Talaván and J. Yáñez, "The quadratic assignment problem. A neuronal network approach", Neural Networks, Vol. 19, 2006, pp. 416-428.
[25] A. K. Tripathi, B. K. Sarker and N. Kumar, "A GA based multiple task allocation considering load", International Journal of High Speed Computing, 2000, pp. 203– 214.
[26] J. D. Ullman, "NP-Complete Scheduling Problems", JCSS, Vol. 10, 1975, pp. 384-93.