Pergamon

Chaos, Solitons & Fractals Vol. 5, No. 5, pp. 739–746, 1995

0960-0779/95$9.50 + .00

0960-0779(94)00175-8

A Natural Model and a Parallel Algorithm for Approximately Solving the Maximum Weighted Independent Set Problem

MOHAMED AFIF

LAMSADE, Université Paris-Dauphine, Place du Maréchal De Lattre de Tassigny, 75775 Paris, Cedex 16, France

and

ARISTIDIS LIKAS

Department of Electrical and Computer Engineering, National Technical University of Athens, 157 73 Zographou, Athens, Greece

and

VANGELIS TH. PASCHOS

LAMSADE, Université Paris-Dauphine, Place du Maréchal De Lattre de Tassigny, 75775 Paris, Cedex 16, France

(*Received* 17 *June* 1994)

Abstract—A dynamical model based upon a physical metaphor is described, and a parallel algorithm inspired from the model is developed for approximately solving maximum weight independent set problem. Our model treats an independent set as an attraction game, where vertices of the graph are considered as still bodies and edges as cells attracted by the still bodies corresponding to its extremities. In addition, we discuss how, by using an analogous model, an approximation algorithm can be developed for the minimum set covering problem.

1. INTRODUCTION

Consider a graph $G = (V, E)$ of order n. An independent set is a subset of $S \subseteq V$ such that there is no pair of nodes in S linked by an edge in G and the maximum independent set problem (IS) is to find an independent set of maximum size. This problem admits a natural generalization, in the case where we associate positive weights (costs) w_i, $i = 1, \ldots, n$, to the vertices of its instance, and then the objective becomes to maximize the sum of the weights of the vertices of an independent set. (In what follows, we denote by WIS the weighted version of the maximum independent set problem.)

WIS constitutes one of the most famous *NP*-complete problems and, in addition, even its constant approximability in polynomial time is *NP*-complete also [1]. For this reason, a number of sequential [2–4] or parallel [5, 6] heuristics have been developed for IS (both the weighted and unweighted cases).

In this paper inspired by [7] where a dynamical system and a parallel algorithm have been developed for the minimum weighted vertex covering problem, we propose a model for WIS. This model, based upon a magnetism metaphor, is described by a dynamical system giving rise to a parallel algorithm approximately solving WIS. Moreover, it seems

that the proposed model is well-adapted to the problem since, as is proved below, the dynamical system converges to an equilibrium state corresponding to a feasible WIS solution.

Moreover, we show that, by extending the proposed approach, it is possible to define an analogous model for approximately solving the minimum set covering problem. Given a collection $\mathcal{S}$ ($|\mathcal{S}| = n$) of subsets of a finite set C ($|C| = m$), a cover is a subcollection $\mathcal{S}' \subseteq \mathcal{S}$ such that $\cup_{S_i \in \mathcal{S}'} S_i = C$, and the minimum set covering problem (SC) is to find a cover of minimum size. By associating weights w_i, $i = 1, \ldots, n$, to the sets of $\mathcal{S}$, we obtain WSC (weighted set covering), a generalization of SC. The objective of WSC is to minimize the total weight of a set covering, this weight being the sum of the weights of the sets in the covering. In [8], Hifi *et al.* propose a Boltzmann machine architecture, treating SC as a particular instance of the minimum vertex covering problem. In this paper, we propose a natural model for WSC fairly similar to the one for WIS, and we prove that the dynamical system describing the WSC model reaches stability; however, as we can see, this model has the drawback of not always providing feasible solutions.

In what follows, given an undirected graph $G = (V, E)$ of order n, we denote by $\Gamma(v_i)$ and w_i the neighbour set and the weight of each vertex v_i ($v_i \in V$); also, we denote by δ_i the cardinality $|\Gamma(v_i)|$ (the degree of v_i).

2. THE DYNAMICAL SYSTEM AND THE DEDUCED PARALLEL APPROXIMATION ALGORITHM FOR THE WEIGHTED INDEPENDENT SET

The natural metaphor we have used to model and solve WIS is the following. Consider the vertex set V as a set of still bodies placed on a bi-dimensional space, and the edge set E as a set of cells located between pairs of still bodies, each cell being likely to be attracted by one of the still bodies that lie on the extremities of the corresponding edge.* Once a vertex† succeeds to attract one of the edges incident to it, then it immediately attracts all of its incident edges and wins in the competition over its neighbours, this victory entailing the inclusion of the winner in the WIS solution and the corresponding exclusion of its neighbours.

A potential $U_i(t)$ is associated to every vertex $v_i \in V$, for any instant t. The motion of a cell c_{ij} (corresponding to the edge v_iv_j), located between v_i and v_j, is described at any instant t by a function $x_{ij}(t)$. It holds that $x_{ij}(t) \to \infty$ whenever c_{ij} is located close to v_j, $x_{ij}(t) \to -\infty$ whenever c_{ij} is located close to v_i, while whenever c_{ij} lies in the middle of the distance between v_i and v_j, $x_{ij}(t) = 0$. Finally, since G is undirected, we consider that the cell c_{ji} is identical to the cell c_{ij}.

Cell c_{ij} moves towards the body $\text{argmax}_{v_iv_j \in E}\{U_i(t), U_j(t)\}$ with velocity

$$\dot{x}_{ij}(t) = U_j(t) - U_i(t), \qquad v_iv_j \in E. \tag{1}$$

Moreover, we define the following transformation $r(t)$ which constraints the values of $x_{ij}(t)$ within the interval $[-1, +1]$:

$$r_{ij}(t) = \tanh(\gamma x_{ij}(t)), \qquad v_iv_j \in E \tag{2}$$

where γ is a positive constant (parameter of the system) that allows the adjustment of the hyperbolic tangent's slope. Based on the above transformation, we have $r_{ij}(t) \to +1$

*Of course, we suppose that the cells are initially located at positions equidistant from the associated bodies; moreover, these distances are finite.

†From here on we shall treat the terms still body and vertex as equivalent; we do the same for the terms cell and edge, respectively.

whenever c_{ij} is located close to v_j, $r_{ij}(t) \to -1$ whenever c_{ij} is located close to v_i, while whenever c_{ij} lies in equal distance from v_i and v_j, $r_{ij}(t) = 0$. Let us note that, since hyperbolic tangent function, tanh, is odd, we have $r_{ij}(t) = -r_{ji}(t)$ for every instant t.

It remains now to define the potential of each vertex. This is defined in such a way that $U_i(t) \in [-w_i, +w_i]$, $v_i \in V$. Let us remark that a natural definition of such potential would be $U_i(t) = (w_i/\delta_i)\sum_{v_j \in V} r_{ji}(t)$, but whenever G contains isolated vertices then the potential of these vertices would be undefined; so, we set

$$U_i(t) = \frac{w_i}{\delta_i + 1}\left[\left(\sum_{v_j \in \Gamma(v_i)} r_{ji}(t)\right) + 1\right] v_i \in V. \tag{3}$$

It is easy to see that in the case where $\delta_i = 0$, $U_i(t) = w_i$.

The system of differential equations induced by expressions (1)–(3) describes the evolution of the dynamical system over time.

Based on these expressions, we can define a WIS solution S in the following way:

$$S = \{v_i \in V: U_i = \lim_{t \to \infty} U_i(t) = w_i\}. \tag{4}$$

Proposition 1. The set S defined by expression (4) is feasible for WIS.

Proof. Let us suppose that there exists an edge $v_i v_j \in E$ such that $\{v_i, v_j\} \subseteq S$. Since $v_i \in S$, $U_i = w_i$ and from expression (3), we get $\sum_{v_j \in \Gamma(v_i)} r_{ji}(t) = \delta_i$; this last expression implies that, for all $v_i v_k \in E$, $r_{ki} = 1$; so, $r_{ji} = 1$. If we follow the same arguments for the fact that $v_j \in S$, we can also find that $r_{ij} = 1$, which is a contradiction since, by the definition of the model, $r_{ij}(t) = -r_{ji}(t)$ for every t. ■

Let us note that the quantity $x_{ij}(t)$ can be written as $x_{ij}(t) = x_{ij}(t_0) + \int_{t_0}^{t} (\mathrm{d}x_{ij}(s)/\mathrm{d}s)\,\mathrm{d}s$. So, by discretization, we obtain

$$x_{ij}(t + \mathrm{d}t) = x_{ij}(t_0) + \mathrm{d}t \sum_{s=t_0(\mathrm{d}t)}^{t} \frac{\mathrm{d}x_{ij}(s)}{\mathrm{d}s}. \tag{5}$$

By means of expressions (1)–(5), we can specify the WIS Algorithm 1. It must be noted that only quantities computed at instant t are necessary for the computations at instant $t + \mathrm{d}t$. Therefore, Algorithm 1 can very easily be implemented on a parallel machine.

It remains now to show that the described dynamical system (and, consequently, Algorithm 1 based upon this system) converges to an equilibrium state. To do this, it is sufficient to show that there is a function $E(t)$ constituting a Lyapunov function for the system or, in other words, admitting the property $\dot{E} \leqslant 0$.

Proposition 2. The function

$$E(t) = \frac{1}{2}\sum_{v_i}\sum_{v_j} r_{ij}(t)\frac{w_i}{\delta_i + 1}\sum_{v_k} r_{ik}(t) + \sum_{v_i}\sum_{v_j}\frac{w_i}{\delta_i + 1} r_{ij}(t)$$

---

begin

fix constants γ, ε, ϵ;
fix a time-step $\mathrm{d}t$;
$t \leftarrow 0$

for all $v_i v_j \in E$ **do**
$x_{ij}(0) \leftarrow \varepsilon$;
$r_{ij}(0) \leftarrow \tanh(\gamma x_{ij}(0))$
od
repeat
for all $v_i \in V$ **do** $U_i(t) \leftarrow (w_i/\delta_i + 1)[(\sum_{v_j \in \Gamma(v_i)} r_{ji}(t)) + 1]$ **od**
for all $v_i v_j \in E$ **do**
$x_{ij}(t + \mathrm{d}t) \leftarrow x_{ij}(t) + \mathrm{d}t(U_j(t) - U_i(t))$;
$r_{ij}(t + \mathrm{d}t) \leftarrow \tanh(\gamma x_{ij}(t + \mathrm{d}t))$;
$t \leftarrow t + \mathrm{d}t$
od
until $|r_{ij}(t) - 1| \leqslant \epsilon$
$S = \{v_i \in V: \lim_{t \to \infty} U_i(t) = w_i\}$
end.

Algorithm 1

constitutes a Lyapunov function for the system described by expressions (1)–(3).

Proof. Let us recall that $r_{ij}(t) = -r_{ji}(t)$, for every t; also $x_{ij}(t) = -x_{ji}(t)$ and, by model definition, $\dot{x}_{ij}(t) = -\dot{x}_{ji}(t)$, $\mathrm{d}r_{ij}/\mathrm{d}x_{ij} = \mathrm{d}r_{ji}/\mathrm{d}x_{ji}$. It holds that $\mathrm{d}E/\mathrm{d}r_{ij} = (w_i/\delta_i + 1)(\sum_{v_i v_k \in E} r_{ik} + 1) = U_i$ (by expression (3)), independently of v_j. Moreover, $\dot{E}(t) = \sum_{v_i} \sum_{v_j, j<i} (\mathrm{d}E/\mathrm{d}r_{ij} - \mathrm{d}E/\mathrm{d}r_{ji})(\mathrm{d}r_{ij}/\mathrm{d}x_{ij})\dot{x}_{ij}(t)$, this expression yielding $\dot{E}(t) = -\sum_{v_i} \sum_{v_j, j<i} (U_j - U_i)^2 (\mathrm{d}r_{ij}/\mathrm{d}x_{ij}) \leqslant 0$. ■

3. EXPERIMENTAL RESULTS

We have tested Algorithm 1 considering two performance criteria: execution time (in seconds) and approximation ratio.[‡] In fact, concerning the latter criterion, we have estimated the quantity $\bar{\rho} = (\sum_I \rho_I)/|\mathcal{I}|$, where ρ_I is the approximation ratio of the algorithm on an IS instance I and $|\mathcal{I}|$ is the number of the produced graphs. Moreover, in order to have some comparative performance information, we have implemented the natural greedy algorithm 2 for IS, and we have tested it following the same performance criteria. Finally, in order to find the optimal solutions of the tested instances, we have implemented a branch-and-bound method adapted to IS.

begin
$S \leftarrow \varnothing$
repeat
$v_i \leftarrow \mathrm{argmax}_{v_j \in V}\{w_j/\delta_j\}$;
$S \leftarrow S \cup \{v_i\}$;
$V \leftarrow V \backslash (\{v_i\} \cup \Gamma(v_i))$
until $V = \varnothing$
end.

Algorithm 2

[‡]In polynomial approximation theory, a usual criterion of the performance of a heuristic on an instance of an *NP*-complete problem is the ratio 'size (value in the weighted case) of the solution provided by the heuristic over size (or value) of the optimal solution of the instance'.

We have generated randomly 161 graphs, 89 among them being regular and 72 irregular. The orders of the generated graphs vary between 15 and 100, and the numbers of their edges between 20 and 200. For the regular graphs, the degrees of the vertices vary between 2 and 7, while for the irregular ones the degrees vary between 1 and 7. Both methods have been executed twice on each instance: one time for the unweighted case and one time by considering integer weights on the vertices, these weights being randomly chosen from the set $\{1, \ldots 30\}$. The tests have been executed on a 33 MHz 486 PC; the execution times for Algorithm 1 concern this sequential implementation.

3.1. *A summary of the performance of the two algorithms*

We present in this section a summary of the performance (concerning average case ratios and average execution times) of the proposed approach (Algorithm 1) and the greedy heuristic (Algorithm 2), respectively. Table 1 summarizes the performance of the two heuristics for the case where all of the weights are equal to 1 (for this case, the selection performed at each step of Algorithm 2 becomes a simple minimum degree vertex choice), while in Table 2, weighted cases are considered.

Since Algorithm 1 uses the parameters ϵ, γ and dt, the performance of the algorithm, concerning both approximation ratios and execution times, depends on the values of these parameters. In this section, we present the parameter values allowing us to obtain good approximation results in reasonable execution times; in Section 3.2, we present results from experiments concerning the influence of these three parameters upon the performance of Algorithm 1.

As one can see from Tables 1 and 2, the experimental behaviour of the conceived model is quite satisfactory and superior to the one of the greedy algorithm, in what concerns the average case approximation ratio. Of course, concerning the corresponding execution times, we note here that we have executed Algorithm 1 sequentially. However, since it is easily parallelizable, its parallel execution time is expected to be much smaller and comparable to the one of the greedy algorithm.

3.2. *Influence of the parameters ϵ, γ and dt on the behaviour of Algorithm* 1

The parameters ϵ (computation's precision), γ (slope of the function tanh) and dt (computation step) used by Algorithm 1 play an important role on both the approximation ratio and the execution time of the algorithm. So, in Tables 3–5, we present experimental

Table 1. Unweighted IS approximation performances of the two heuristics; Algorithm 1 has been executed with $\epsilon = 10^{-8}$, $dt = 10^{-3}$ and $\gamma = 250$

Method	$\bar{\rho}$	% Optimal solutions	Average execution time
Algorithm 1	0.947	57.14	10.8
Algorithm 2	0.918	41.61	0.77

Table 2. Weighted IS approximation performances of the two heuristics; Algorithm 1 has been executed with $\epsilon = 10^{-8}$, $dt = 10^{-4}$ and $\gamma = 200$

Method	$\bar{\rho}$	% Optimal solutions	Average execution time
Algorithm 1	0.963	39.75	12.8
Algorithm 2	0.959	36.65	1.02

Table 3. Performance measures of Algorithm 1 under variation of ϵ, with $dt = 10^{-3}$ and $\gamma = 200$

ϵ	10^{-4}	10^{-6}	10^{-8}
		Unweighted case	
% Optimal solutions	35.40	40.99	53.42
$\bar{\rho}$	0.896	0.915	0.942
Mean execution time	4.7	7.6	13.5
		Weighted case	
% Optimal solutions	28.57	31.68	31.68
$\bar{\rho}$	0.95	0.954	0.954
Mean execution time	1.0	2.2	2.9

Table 4. Performance measures of Algorithm 1 under variation of dt, with $\epsilon = 10^{-8}$ and $\gamma = 200$

dt	10^{-1}	10^{-2}	10^{-3}
		Unweighted case	
% Optimal solutions	28.57	41.61	53.42
$\bar{\rho}$	0.896	0.933	0.942
Mean execution time	1.6	5.2	13.5
		Weighted case	
% Optimal solutions	11.80	18.01	31.68
$\bar{\rho}$	0.87	0.904	0.954
Mean execution time	0.2	1.2	2.9

Table 5. Performance measures of Algorithm 1 for 3 under variation of γ, with $dt = 10^{-3}$ and $\epsilon = 10^{-8}$

γ	100	150	200	250
			Unweighted case	
% Optimal solutions	41.61	45.34	53.42	57.14
$\bar{\rho}$	0.912	0.926	0.942	0.947
Mean execution time	17.1	17.0	13.5	10.8
			Weighted case	
% Optimal solutions	33.54	33.54	31.68	28.57
$\bar{\rho}$	0.958	0.956	0.954	0.950
Mean execution time	3.1	3.0	2.9	2.4

results on the dependence of these measures with respect to the variations in the values of the parameters ϵ, dt and γ, respectively.

As can be observed, by performing judicious choices of the three parameters, it is possible to achieve a very efficient trade-off between solution qualities and execution times for Algorithm 1.

4. A MODEL FOR MINIMUM SET COVERING PROBLEM

A very convenient way to represent SC is by means of a bipartite graph $B = (S, C, E)$, where the colour class S represents the members of $\mathcal{S}$, the colour class C represents the

elements of C, and the edge set E contains an edge $s_i c_j$, if $c_j \in S_i$ in the original SC instance; in the case of WSC, the vertices of S are weighted.

To each edge $s_i c_j$ of E, we correspond a real input value $r_{ij}(t)$ and an output value $v_{ij}(t) = \tanh(r_{ij}(t))$. Moreover, for every vertex s_i, we define, as for the case of IS, a quantity p_i which we call the potential of vertex s_i and define it as $p_i = (\sum_{j=1}^{|S_i|} r_{ji}(t) - w_i)/|S_i|$.

We have developed the following approximation algorithm for SC.

STEP 0 set $r_{ij}(t) = 0$ and fix positive constants γ and ϵ; at each time step t do
STEP 1 compute $p_i(t)$, $i = 1, \ldots, n$;
STEP 2 for every c_j, $j = 1, \ldots, m$, find the vertices $s_j^{\max}$ and $s_j^{\min}$, adjacent to c_j, having the maximum ($p_j^{\max}(t)$) and the minimum ($p_j^{\min}(t)$) potential, respectively, over all vertices adjacent to c_j;
STEP 3 for every $j = 1, \ldots, m$ and for every k such that s_k is adjacent to c_j, set $\Delta r_{jk}(t) = \gamma(p_j^{\max}(t) - p_k(t))$, if $s_k \neq s_j^{\max}$ and $\Delta r_{jk}(t) = b_j(t)(p_j^{\max}(t) - p_j^{\min}(t))$, if $s_k = s_j^{\max}$, where γ is a small positive constant and $b_j(t)$ is a parameter which must be computed at each time step in order to ensure convergence (the role of $b_j(t)$ is discussed below);
STEP 4 compute $r_{jk}(t+1) = r_{jk}(t) + \Delta r_{jk}(t)$ and $v_{jk}(t+1) = \tanh(r_{jk}(t+1))$;
STEP 5 if $|v_{jk}(t+1) - v_{jk}(t)| \leq \epsilon$, for all j, k then exit, else go to STEP 1.

It is easy to see that at each time step, for every edge i, $i = 1, \ldots, m$, there exists a j^i for which $v_{ij^i}(t)$ increases while, for all $j \neq j^i$, the values of v_{ij} decrease. To prove convergence of the above procedure, it suffices to show that there exists a function $L(t)$ bounded from below and decreasing with t; such a Lyapunov function for the system is $L(t) = \sum_i \sum_j v_{ij}(t)$.

Since $|v_{ij}| < 1$, $L(t)$ is bounded below.

To prove that $L(t)$ decreases with t, we impose the requirement that, for each element c_i, $\delta v_{ij^i}(t+1) < \sum_{j, j \neq j^i} |\delta v_{ij}(t+1)|$, where $\delta v_{ij}(t+1) = v_{ij^i}(t+1) - v_{ij^i}(t)$. This inequality can be easily satisfied by first computing the corresponding sum, and then by appropriately adjusting the parameter $b_i(t)$ (appearing in STEP 3 of the above algorithm) so that the inequality becomes valid.

Under the above operation schema, the system asymptotically converges and moreover, as one can see, the only equilibrium point is the one having the property: 'for each c_i, $v_{ij^i} = 1$ and $v_{ij} = -1$, $j \neq j^i$'.

In terms of the proposed algorithm, this means that, if a vertex s_j has 'won' in the competition for at least one c_i (at least one $v_{ij} = 1$), then the set S_j corresponding to vertex s_j is included to SC solution.

The described algorithm does not always lead to feasible solutions. We have tested it on 150 (unweighted) SC instances where n varied between 8 and 20, m between 10 and 30, the cardinality of a set of $\mathcal{S}$ varying between 2 and 15; the parameters γ and ϵ were both fixed to 10^{-3}.

The percentage of feasible solutions was quite high, equal to 76.34%, the average approximation ratio (considering the instances for which feasible solutions had been obtained) was 2.22, while the percentage of the optimal solutions obtained was equal to 2%.

5. CONCLUSIONS

We have defined a dynamical model inspired from a physical metaphor that leads to a parallel approximation algorithm for WIS and, moreover, we have provided some, not yet

completely elaborated, ideas of how the proposed approach could be extended to develop a parallel algorithm for approximately solving WSC. (We are actually working on the conception of an 'efficient' dynamical system for WSC.) Concerning the IS Algorithm 1, the tests we have performed show that this algorithm is very efficient from an approximation quality point of view. In addition, the sequential execution time of Algorithm 1 is quite satisfactory, and we can consider that, when implementing this algorithm on a parallel machine, the total execution time will be even smaller than the corresponding time of the greedy heuristic.

REFERENCES

1. S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy, Proof verification and intractability of approximation problems, in *Proc. FOCS* '92, (1992) pp. 14–23.
2. D. S. Hochbaum, Efficient bounds for the stable set, vertex cover and set packing problems, *Discr. Appl. Maths*, **6**, 243–254 (1983).
3. V. Th. Paschos, A $\Delta/2$-approximation algorithm for the maximum independent set problem, *Information Processing Letters*, **44**, 11–13 (1992).
4. V. Th. Paschos and M. Demange, A heuristic for maximum independent set problem with improved approximation ratio, manuscript (1994).
5. E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines, A Stochastic Approach to Combinatorial Optimization and Neural Computing*. John Wiley & Sons Ltd (1989).
6. V. Th. Paschos, F. Pekergin and V. Zissimopoulos, Approximating the optimal solutions of some hard graph problems by a Boltzmann machine, *JORBEL*, Special Issue on Combinatorial Optimization in Applications, **33**, 119–132 (1993).
7. A. Likas and A. Stafylopatis, A parallel algorithm for the minimum weighted vertex cover problem, *Information Processing Letters*, to appear.
8. M. Hifi, V. Th. Paschos and V. Zissimopoulos, A new efficient sequential heuristic and a neural network for minimum set covering problem, manuscript (1994).