On-Line Graph Searching by a Smell-Oriented Vertex Process

Israel A. Wagner 1,2, Michael Lindenbaum 2, and Alfred M. Bruckstein 2,3

1 IBM Haifa Research Lab, Matam, Haifa 31905, Israel
2 Department of Computer Science, Technion City, Haifa 32000, Israel
3 AT&T Bell Labs at Murray-Hill, NJ 07974, USA
israelw@vnet.ibm.com, {miclfreddy}@cs.technion.ac.il

Abstract
An ant walks along the edges of a graph G, occasionally leaving pheromone traces at vertices, and using those traces to guide its exploration. We show that the ant can cover the graph within time $O(nd)$ where $n$ is the number of vertices and $d$ the diameter of G. The use of traces achieves a trade-off between random and self-avoiding walks, as it can give lower priority to already visited neighbors. A Hamiltonian cycle in G, if one exists, is a limit cycle of the smell directed graph exploration process.

Introduction
Following an ancient advice 1, we consider a memoryless agent that searches a graph G for food. The ant has the ability to leave pheromone traces on vertices, and to sense the smell traces at the current location and its immediate neighbors. By "sensing the smell" at a vertex we mean that the ant knows both the number of smell traces that have been left on the vertex, and the time of the most recent trace. Formally, a vertex $v$ at time $t$ is marked by a pair $(\sigma_t(v), \tau_t(v))$ where $\sigma_t(v)$ is the number of marks left on $v$, and $\tau_t(v)$ is the time of the most recent mark left there up to time $t$. Initially we set both marks to zero for all $v \in V$, $V$ being the set of vertices. Being at a vertex $u$, the ant smells around and chooses among $N(u)$, the set of neighbors of $u$, neighbor $v$ with a mark lower than itself, i.e. such that $(\sigma_t(v), \tau_t(v)) < (\sigma_t(u), \tau_t(u))$, where "<" means "less than" in lexicographic order. If no such vertex exists in the neighborhood of $u$ (i.e. $u$ is a local minimum of $\sigma_t(.)$), the ant increases $\sigma_t(u)$ by 1, and sets $\tau_t(u)$ to the current time. Intuitively, this rule of motion behaves like a steepest-decent optimizer, with the additional option to dynamically alter the cost function, thereby avoiding being stuck in a local minima.

We shall call this process "vertex ant walk" to distinguish it from a similar "edge ant walk" process discussed in (Wagner, Lindenbaum & Bruckstein 1996) where the edges, rather than the vertices, were marked.

1 "Go to the ant. thou sluggard; consider her ways, and be wise" (Proverbs. vi. 6).

It has been shown there that a group of $k$ smell-oriented ants that evolve in an edge process can cover a graph in a (worst case) time of $O((\Delta n^2)/k)$, where $\Delta$ is the maximum vertex-degree and $n$ - the number of vertices. The result in the current paper is an improvement since, in general, $nd < \Delta n^2$.

The VAW process, beyond its theoretical interest, has applications in robotics where a robot with limited sensing capabilities but with the ability to leave marks on the ground has to cover a closed region for purposes of cleaning a dirty floor, painting a wall, or demining a mine-field. Another potential application is searching a large network of WWW sites which are changing slowly; if a web-site does not change more than once in $T$ units of time, and $T < 2dn$, then our method guarantees that no change will be missed.

Related work: Graph search is an old problem; several methods exist for deterministic (e.g. (Even 1979), (Fraenkel 1970), (Hopcroft & Tarjan 1973), (Tarry 18..), (Tarjan 1972)) random (e.g. (Aleliunas et al. 1979), (Barnes & Feige 1993), (Broder et al. 1994)) and semi-random ((Gal & Anderson 1990)) covering. A step towards a trace-oriented theory of search was done in (Blum & Sakoda 1977) and (Blum & Kozen 1978), where pebbles are used to assist the search. Pebbles are tokens that can be placed on the floor and later be removed. In (Blum & Kozen 1978) it was shown that a finite automaton with two pebbles can search all mazes. In a sense, our work is a generalization of this work, since one may use "diminishing pebbles" or "deflating tokens" as a model of odor markings. There are several related search methods in the literature, some of which will be briefly mentioned in the sequel. The RTA* (Real-Time A*) and LRTA* (Learning RTA*) (Korf 1990) are two variations on the famous A* heuristic search, with a cost function that takes the current searcher's location into account, thus making the algorithm more realistic for field applications like robotics. In (Thrun 1992),(Thrun 1992a) a counter-based exploration method is described which is quite similar to our method, but the upper bound shown there on the cover time is $O(n^2d)$, where $d$ is the diameter and $n$ is the number of vertices in the graph.
bor Approach (NNA) method of (Koenig & Smirnov 1996), a graph is learned by moving towards the nearest un-visited edge in the graph, and an upper bound is proved on the cover time of $O(m \log n)$ where $m$ is the number of edges in the graph. The usage of ideas from nature for search and optimization problems has recently acquired popularity. See (Dorigo, Maniezzo & Colorni 1996) and references therein for an ant-system used to solve Travelling Salesperson (TSP) problems.

Our algorithm suggests a reasonable trade-off between the rigid, highly sensitive Depth-First Search (DFS) and self avoiding (Madras & Slade 1993) walk on one hand, and the absolutely adaptive (but very time-consuming) random walk, on the other hand.

In this extended abstract we prove an $O(nd)$ bound on the cover time of a graph by the above process, where $n = |V(G)|$ and $d = \text{diam}(G)$, and show that a Hamiltonian cycle, if one exists in $G$, is a limit cycle of the process.

Vertex Ant Walk - a Tradeoff Between Random and Self-Avoiding Walks

Formally, the process is defined by the following rule of motion:

```
Rule Vertex-Ant-Walk(u vertex;
A) v := u's neighbor with
minimal value of $(\sigma(\cdot), \tau(\cdot))$;
   (if there is a tie - make a random decision)
B) if $\sigma(u) \leq \sigma(v)$ then
C) $\sigma(u) := \sigma(u) + 1$;
D) $\tau(u) := t$;
E) $t := t + 1$;
F) go to v.
end Vertex-Ant-Walk.
```

In the sequel we shall refer to the above rule as the VAW rule. See Figure 1 for an example of the VAW evolution.

How efficient is this process? Let us now show that it covers any connected graph within $O(nd)$ steps, where $n$ is the number of vertices and $d$ is the diameter of the graph.

**Lemma 1** If $(u, v) \in E$ then it always holds that $|\sigma(u) - \sigma(v)| \leq 1$.

**Proof** The lemma is clearly true when $t = 0$ since all $\sigma(\cdot)$ values are being preset to zero. Assuming it is also true at time $t$, we claim that the $(t + 1)$th step of VAW does not cause any harm, i.e. the assertion of the lemma remains true. Note that the only change in $\sigma$ at this time may occur at the current node $u$ in step C of the VAW rule. If the ant finds that $\sigma(u) > \sigma(v)$ then there is no change to $\sigma$ and the assertion remains true. Otherwise, $\sigma(u) \leq \sigma(v)$ and $\sigma(u)$ is increased, but this again causes no harm since $\sigma(u)$ is a minimum among $N(u)$, hence incrementing its value by one cannot cause it to differ by more than 1 from any other neighbor.

Our next step is to show that under the VAW rule, vertices are marked in at least 1/2 of the steps. For this end, let us define the total smell on the graph at time $t$, $\sigma_t(G)$, to be the sum of all the $\sigma$ values on the vertices at that time.

**Lemma 2** At all times $t$,

$$\sigma_t(G) \geq \sum_{u \in V} \sigma_t(u) \geq t/2.$$  

**Proof** Let us denote the sequence of nodes visited up to time $t$ by $x_1, x_2, \ldots, x_t$. According to the VAW rule, upon moving from $x_i$ to $x_{i+1}$, it holds that either $\sigma(x_{i+1}) < \sigma(x_i)$ (i.e. a negative gradient existed at time $i$) or $\sigma(x_{i+1}) \leq \sigma(x_i)$ (i.e. the ant created an anti-gradient by increasing $\sigma(x_i)$). It is also clear that

$$\sigma(x_1) = [\sigma(x_1) - \sigma_{t-1}(x_{t-1})] + \ldots + [\sigma(x_t) - \sigma(1) + \sigma(1).$$

But $\sigma(x_1) = 0$, hence

$$\sigma(x_1) = \sum_{i=2}^{t} \delta_i,$$

where $\delta_i = \sigma(x_i) - \sigma_{i-1}(x_{i-1})$. According to Lemma 1, each $\delta_i$ is either $0$, $0$, or $1$. In the last two cases, $\sigma_i(x_i)$ should have been increased in the $(i+1)$th step, otherwise the ant could not move to $x_{i+1}$ - recall that it is only allowed to move along a negative gradient of $\sigma(\cdot)$. Denote by $t_n, t_0$ and $t_p$ the number of $\delta_i$'s, up to time $t$, with values $-1, 0$, and 1, respectively. Clearly, $t = t_n + t_0 + t_p$ and $\sigma(G)$, the total sum of $\sigma$ over $G$, is exactly the number of increases in $\sigma(\cdot)$, that is $\sigma(G) = t_p + t_0$. On the other hand, observe that $t_p - t_n \geq 0$ since

$$t_p - t_n = \sigma(x_t) \geq 0,$$

which implies $t_p \geq t_n$ and hence $t_p + t_0 \geq t/2$, and

$$\sigma(G) = t_p + t_0 \geq t/2.$$  

Now let us combine the continuity of $\sigma(\cdot)$ (Lemma 1) together with its temporal accumulation (Lemma 2) to get

**Theorem 1** Denote by $d$ the diameter of $G$, and by $n$ - the number of vertices. Then $\sigma(2n-d-1+1(u) > 0$ for all $u \in V$. In other words - after at most $d(2n - d - 1) + 1$ steps the graph is covered.
Proof Assume that at time $t$ the graph has not yet been fully covered. Then there exists at least one node, say $u$, for which $\sigma(u) = 0$. According to Lemma 1, none of $u$'s neighbors can have $\sigma > 1$, none of the neighbors' neighbors can have $\sigma > 2$, etc. Hence we have at least one node with $\sigma = 0$, at least one with $\sigma = 1$, and so on up to $d-1$. The rest of the nodes (at most $n - d$) may have $\sigma \leq d$. This gives a total $\sigma(G)$ of at most $d(d-1)/2 + (n-d)$.

But according to Lemma 2, $\sigma$ is increased in at least half the steps; hence if $t > d(2n - d - 1)/2$ the graph is necessarily covered.

The bound of $O(nd)$ is nearly tight; the example in Figure 2 depicts a case where the cover time required by VAW is indeed $O(nd)$.

**VAW and the Hamilton cycle**

As opposed to DFS and similar search methods, our VAW ant never stops, unless we pre-program it with an upper bound on its walking time. Hence, VAW is not the best method for a one-shot search; however, as we shall show in this section, it may be a competitive candidate for an adaptive-repetitive search, in which we want the graph to be searched again and again. Applications may be guarding/surveillance or keeping an up-to-date database in the context of a huge network of data sources (e.g. INTERNET).

Let us assume that after covering the graph we allow our ant to continue its YAW process. It follows from Theorem 1 that once every (at most) $2dn$ steps, another coverage of the graph is completed. If the graph has a Hamiltonian cycle, following this cycle is clearly the shortest way to cover the graph. We shall now show that a Hamiltonian cycle (if one exists in $G$) is a limit cycle of the VAW process; i.e. if the ant happens to follow such a cycle for one time, it will follow this cycle forever.

**Lemma 3** A Hamiltonian cycle in $G$ is a limit cycle of the VAW process.

**Proof:** Let us denote by $t_0$ the time of completion of a Hamiltonian path. Hence, according to our assumption, the $n$ vertices visited prior to $t_0 + 1$ exactly constitute the set $V$, and the following steps complete the cycle, i.e. $x_{t_0 + 1} = x_{t_0 - n + 1}$. By the VAW rule, an ant can only follow the anti-gradient of the $(\sigma(.), \tau(.))$ function; hence, at time $t_0$, it should hold that $\sigma_{t_0}(x_{t_0-n+1}) \geq \sigma_{t_0}(x_{t_0-n+2}) \geq \cdots \geq \sigma_{t_0}(x_{t_0})$. But then we know that at time $t_0 + 1$ our ant goes again to $x_{t_0-n+1}$, which implies that $\sigma_{t_0+1}(x_{t_0}) > \sigma_{t_0}(x_{t_0-n+1})$. But this is only possible if $\sigma$ values on all vertices increase exactly by one during each cycle. Hence when the ant visits $x_{t_0+1}$, all its neighbors (with the exception of $x_{t_0}$) have $\sigma$ value equal to that of $x_{t_0+1}$; hence $\sigma$ is increased by 1 for that vertex, and the next vertex is chosen according to $\tau$, the secondary parameter. But this parameter holds, for each neighbor, the time of the latest change done to its marking - hence the oldest neighbor is chosen and the cycle continues on and on.

**Some Open Questions**

The vertex ant walk seems to be quite a simple process; however, several facets of its behavior are still challenging, as is shown by the following examples.

- Does a VAW process on a Hamiltonian graph always converge to a limit cycle? In other words, are there any non-Hamiltonian limit cycles for such a graph? In general, does the spanning tree induced by VAW improve (in terms of covering cycle) with time?

- A probabilistic version of VAW rule does not determine the next neighbor specifically, but assigns each neighbor a probability according to its current $(\sigma, \tau)$ mark, e.g. the probability of jumping from $u$ to $v$ may be

$$
\text{Prob}(u \rightarrow v) = \frac{1}{\sum_{u \in N(v)} 1/(1 + \sigma(u))}.
$$

Is such a semi-random coverage process faster, on the average, or slower than the deterministic one?

**Acknowledgements**

We thank Bob Holt of AT&T-Bell Labs for the careful reading of this paper and for several suggestions for improvements.

**References**


M. Blum, D. Kozen, "On the power of the compass, or, Why mazes are easier to search than graphs," in FOCS'78, p. 124-142.


Figure 1: A VAW ant covers a graph and arrives at a Hamiltonian limit cycle. The corresponding ($\sigma$, $\tau$) evolution is shown in the table. Note that $\sigma(2)$ (at times 4 and 18) and $\sigma(4)$ (at time 12) do not increase since they are not a local minima.

Figure 2: A hard case for the VAW rule. There are n vertices, and about 1.25n edges. The diameter is about 0.8n, and the time needed to traverse it may be as long as $O(nd) = O(n^2)$. The dotted arrows show the worst case where each triangle of vertices is a "trap" that causes the ant to go back to its starting point.