A running time analysis of an Ant Colony Optimization algorithm for shortest paths in directed acyclic graphs

Nattapat Attiratanasunthron ¹ Jittat Fakcharoenphol *,¹

Department of Computer Engineering, Kasetsart University, Bangkok, 10900, Thailand

Abstract

In this paper, we prove polynomial running time bounds for an Ant Colony Optimization (ACO) algorithm for the single-destination shortest path problem on directed acyclic graphs. More specifically, we show that the expected number of iterations required for an ACO-based algorithm with \( n \) ants is \( O\left(\frac{1}{\rho} n^2 m \log n\right) \) for graphs with \( n \) nodes and \( m \) edges, where \( \rho \) is an evaporation rate. This result can be modified to show that an ACO-based algorithm for One-Max with multiple ants converges in expected \( O\left(\frac{1}{\rho} n^2 \log n\right) \) iterations, where \( n \) is the number of variables. This result stands in sharp contrast with that of Neumann and Witt, where a single-ant algorithm is shown to require an exponential running time if \( \rho = O(n^{-1-\epsilon}) \) for any \( \epsilon > 0 \).

Key words: Analysis of algorithms, graph algorithms, Ant Colony Optimization, shortest paths

1. Introduction

Ant Colony Optimization (ACO) is now one of standard paradigms for designing optimization algorithms (see the book by Dorigo and Stützle [3]). The framework employs many agents, called ants, each independently finding feasible solutions with a shared communication mechanism based on pheromone values.

The algorithm works in iterations, each starting in when a set of ants gather information, following by the information exchange phase, when ants update the pheromones according to how good the solutions they see. Since ACO is an iterative algorithm, we measure the running time in the number of iterations.

Recently, Gutjahr [4], and Stützle and Dorigo [8] prove the convergence results, i.e., they show that some forms of ACO eventually find the optimal solutions. However, the numbers of steps in their analyses are exponentially large, since they consider the algorithm in the limit. A more precise analysis of the running times of the ACO-type algorithms is left as an important open problem, where an analysis of one specific problem, called One-Max, is explicitly called for, e.g., in Dorigo and Blum’s survey [2]. (See Section 3.1, for a formal definition of One-Max and its ant-based algorithm.)

Neumann and Witt [6,7] are the first to analyze the running time bounds of various ACO-based algorithms. In [7], they analyze the running time of an ACO-based algorithm for One-Max. They consider an algorithm, called 1-ANT, that uses a single ant to construct solutions, and present a phase transition result on the running time as a parameter of

¹ Research supported by Kasetsart University Research and Development Institute (KURDI).

* Corresponding author.

Email addresses: g4665288@ku.ac.th (Nattapat Attiratanasunthron), jittat@gmail.com (Jittat Fakcharoenphol)
the evaporation rate $\rho$. If the number of variables in One-Max is $n$, they show with high probability that if $\rho = O(n^{-1-\epsilon})$ for some $\epsilon > 0$, the running time of the algorithm is $2^{O(n^{1/2})}$, and if for some $\rho = \Omega(n^{-1+k})$, the running time of the algorithm is $O(n^k)$. Neumann and Witt [6] also analyze the running time of the 1-ANT algorithm to construct minimum spanning trees.

1.1. Results

The main result for this paper is the running time analysis of a multiple-ant algorithm for finding single destination shortest paths in directed acyclic graphs (DAGs). This variant is quite similar to AntNet, an ACO-based routing algorithm proposed by Di Caro and Dorigo [1], except that the pheromone update is performed locally. We show that the expected running time of this algorithm is $O(1/\rho mn^2 \log n)$ for DAG with $n$ nodes and $m$ edges. We note that our analysis is similar to the analysis of the Bellman-Ford algorithm.

The result on the shortest path algorithm on a DAG directly implies that the expected number of iterations required for the algorithm on One-Max is $O(\frac{1}{\rho} n^3 \log n)$, through a simple reduction of Neumann and Witt. However, if the graph is bounded degree, or the number of edges in any shortest paths is bounded, we can modify the previous analysis slightly to get a better bound of $O(\frac{1}{\rho} mn \Delta D \log(\Delta D))$ where $\Delta$ denotes the maximum degree and $D$ denotes the maximum number of edges on any shortest paths. This improves the bound on One-Max to $O(\frac{1}{\rho} n^2 \log n)$ expected iterations, because the maximum degree in the graph after the reduction is 2. Our result implies that if $\rho$ is only polynomially large, i.e., $\rho = \Omega(n^{-k})$, for some fixed $k$, the algorithm runs in expected polynomial time. This is in contrast with Neumann and Witt analysis of a single ant where there is a phase transition at $\rho = n^{-1}$.

Both results rely heavily on the lower and upper bounds of the pheromone values, as in Neumann and Witt.

We give the proof of the main result in Section 2. The result on graphs with degree bounds or bounds on the number of edges on shortest paths is described in Section 3. We mention the running time of the algorithm on One-Max in Section 3.1. Finally, in Section 4, we discuss the difficulties in applying this result to graphs containing cycles.

2. Multiple ants on a DAG

We are given a weighted directed acyclic graph $G = (V,E)$ with length $t$ on the edges, and a destination node $t$. Let $n = |V|$ and $m = |E|$. For path $P$, the length of $P$ is the sum of the lengths of all the edges in $P$. The shortest path from $v$ to $t$ is the path with the shortest length. In this problem, we want to find, for each $v \in V$, the shortest path from $v$ to $t$. It is known that there exists a solution in the form of shortest path trees. We assume that $t$ is reachable from every nodes.

We first note that since $G$ is acyclic, one can always find shortest paths for any real length functions $t$. In general graph, this might not be true, since shortest paths do not exist if some negative-length cycle is present. Also, finding shortest simple paths, in general graph, is known to be NP-hard.

We describe an ACO-based algorithm, called n-ANT, for this problem. Each edge $e = (u,v) \in E$ maintains pheromone value $\tau_e = \tau(u,v)$. Denote by $\deg(u)$ the out-degree of $u$. The initial value of $\tau(u,v) = 1/\deg(u)$. This ensures that initially the sum of the pheromones on the outgoing edges of any node, except the destination $t$, is 1.

The algorithm uses $n$ ants: $a_u$'s for all $u \in V$. Each ant $a_u$ tries to solve the shortest path problem from $u$ to $t$. It also keeps the best shortest distance $f_u$ from $u$ to $t$ and the corresponding current shortest path $\hat{s}_u$.

For each iteration, each ant $a_u$ starts its exploration from node $u$. It then proceeds in steps. If $a_u$ is not at $t$, it chooses one outgoing edge from its current location according to the pheromones. Suppose that it is at node $v$, edge $(v,w)$ is chosen with probability

$$\frac{\tau(v,w)}{\sum_{(v,x) \in E} \tau(v,x)},$$

and if it is chosen, $a_u$ moves to $w$ in the next step. The ant repeats until it reaches the destination $t$. On its way, the ant keeps updating the current path $s_u$ and after it reaches $t$, it computes $f_u$, the length of $s_u$.

After all ants finish, each ant must have one path $s_u$ from $u$ to $t$ of length $f_u$. We then update the pheromones. Each ant is responsible for updating the pheromones on edges leaving its starting nodes. We follow the rule by Neumann and Witt, which is a slight rephrasing of the "best-so-far" update rule in Dorigo and Blum [2]. Ant $a_u$ compares the values of $f_u$ and $f_t$, and if $f_u \leq f_t$, it updates $f_u$.
if \( f_u \leq f'_u \) then
\[
\tilde{f}_u \leftarrow f'_u, \quad \tilde{s}_u \leftarrow s_u
\]
for each \( e = (u, w) \in E \) do

// normal update
\[
\tau_e \leftarrow \begin{cases} 
(1 - \rho)\tau_e + \rho, & \text{if } e \in s_u \\
(1 - \rho)\tau_e, & \text{otherwise}
\end{cases}
\]
// boundary condition
if \( \tau_e > \tau_{\text{max}} \) then, \( \tau_e \leftarrow \tau_{\text{max}} \) (*)
if \( \tau_e < \tau_{\text{min}} \) then, \( \tau_e \leftarrow \tau_{\text{min}} \)

Fig. 1. Pheromone update for ant \( a_u \).

and the pheromones on all edges \((u, w) \in E\). We maintain that the minimum pheromone is at least \( \tau_{\text{min}} = \frac{1}{\rho} \) and the maximum pheromone is at most \( \tau_{\text{max}} = \frac{n^2 - 1}{\rho} \). The update procedure for ant \( a_u \) is shown in Figure 1. After the pheromone update, the algorithm then repeats for the next iteration.

For simplicity, in our analysis, we ignore every node with a single outgoing edge. These nodes are in the graph essentially to ensure that the graph is simple.

We first claim that the sum of the pheromones on the outgoing edges of any node \( u \) is at least 1. To see this, note that initially the sum is one. If during the pheromone update, there is no edges with pheromone value greater than \( \tau_{\text{max}} \), i.e., the then-clause in line (*) is not executed, the sum remains unchanged. If the pheromones on two or more edges get decreased to \( \tau_{\text{max}} \) in line (*), clearly the sum is at least \( 2 \frac{n^2 - 1}{\rho} \geq 1 \), if \( n > 1 \). Now, assume that there exists only one edge \( e \) where \( \tau_e > \tau_{\text{max}} \) after the normal update. Since \( u \) has at least two outgoing edges, there exists another outgoing edge \( e' \) whose pheromone is at least \( 1/n^2 \). Thus, the sum of the pheromones on outgoing edges of \( u \) is at least 1.

The next lemma states that the probability of choosing any edge is not too small.

**Lemma 1** Any edge \( e = (u, v) \) is chosen by any ant at node \( u \) with probability at least \( \Omega \left( \frac{1}{\rho^2 \deg_u(u)} \right) \).

**PROOF.** The probability of choosing \((u, v)\) is
\[
\frac{\tau(u, v)}{\sum_{(u, x) \in E} \tau(u, x)} \geq \frac{\tau_{\text{min}}}{\deg_u(u)\tau_{\text{max}}} = \frac{1/n^2}{(\deg_u(u)(n^2 - 1))/n^2} \geq \frac{1}{\deg_u(u)n^2},
\]
as required. \( \square \)

To prove the running time bound, we use the following notion of progress.

We call an edge \((u, v)\) an incorrect edge if it does not belong to any shortest path from \( u \) to \( t \). We say that node \( u \) is processed if \( \tilde{f}_u \) equals to the shortest distance from \( u \) to \( t \) and the pheromone on every incorrect edge is \( \tilde{\tau}_{\text{min}} \).

Whenever an iteration ends and \( f_u \) equals the shortest distance from \( u \) to \( t \), we say that \( a_u \) sees its optimal solution. The next lemma states the bound on the number of times ant \( a_u \) needs to see the optimal solution before \( u \) is processed.

**Lemma 2** After \( a_u \) sees the optimal solution for at most \( O \left( \frac{1}{\rho} \log \frac{\tau_{\text{max}}}{\tau_{\text{min}}} \right) \) times, node \( u \) is processed.

**PROOF.** Note that after \( a_u \) finds the shortest path, the pheromones of any incorrect edges will not increase. Since then, it gets decreased by a factor of \((1 - \rho)\) for every time \( a_u \) finds any shortest paths. Consider any incorrect edge \( e \). After \( l \) updates, the pheromone on \( e \) would be \( \tau(1 - \rho)^l \), if the pheromone of \( e \) in the first iteration that the shortest path is found is \( \tau \). If we let \( l = \frac{1}{\rho} \ln(\frac{\tau_{\text{max}}}{\tau_{\text{min}}}) \), we have
\[
\tau(1 - \rho)^l \leq \tau_{\text{max}}(1 - \rho)^l = \tau_{\text{max}}(1 - \rho)^{\frac{1}{\rho} \ln(\frac{\tau_{\text{max}}}{\tau_{\text{min}}})} \\
\leq \tau_{\text{max}}(e^{-\rho})^{\frac{1}{\rho} \ln(\frac{\tau_{\text{max}}}{\tau_{\text{min}}})} \\
= \tau_{\text{max}}e^{-\ln(\frac{\tau_{\text{max}}}{\tau_{\text{min}}})} = \tau_{\text{min}}.
\]

Thus, the lemma follows. \( \square \)

We are ready to prove our main theorem.

**Theorem 3** The expected number of iterations for algorithm \( n \text{-ANT} \) for finding single-destination shortest paths on a DAG with \( n \) nodes and \( m \) edges is \( O \left( \frac{1}{\rho} n^2 m \log n \right) \).

**PROOF.** We first bound the expected number of iterations needed to process node \( u \), provided that every node reachable from \( u \) to \( t \) has already been processed.

We show that for any iteration, the probability that \( a_u \) sees its optimal solution is at least \( \Omega(\frac{1}{\rho^2 \deg_u(u)}) \). Before we continue, we note the basic fact on shortest paths. Suppose that node \( v \) is on some shortest path \( P \) from \( u \) to \( t \). Let \( P' \) be a subpath of \( P \) starting at \( u \) and ending at \( v \). For any shortest path \( P_v \) from \( v \) to \( t \), we have that the combined path \( P' \cup P_v \) is also a shortest path from \( u \) to \( t \).
Note that ant $a_u$ has to make at most $n$ random choices to reach $t$. From Lemma 1, it chooses the first edge on the shortest path correctly with probability at least $\Omega \left( \frac{1}{n^2 \deg_o(u)} \right)$. After that, it has to make at most $n - 1$ choices. Ant $a_u$, reaching node $v$, does not follow any shortest paths from $v$ to $t$ only when it chooses some incorrect outgoing edge of $v$. Since all nodes reachable from $u$ to $t$ are processed, the pheromones on every incorrect edge leaving $v$ is $\tau_{\min}$. Thus, the sum of the pheromones on the wrong edges is $\deg_o(v) \tau_{\min} \leq n \cdot \frac{1}{n} = 1/n$. Because the sum of the pheromones on all outgoing edges is always at least 1, the probability that $a_u$ chooses some incorrect edge is at most $1/n$. Thus, $a_u$ continues to make a correct choice with probability at least $(1 - 1/n)$.

Therefore, the probability that $a_u$ sees its optimal solution is at least $(1 - 1/n)^{n - 1} \cdot \Omega \left( \frac{1}{n^2 \deg_o(u)} \right) = \Omega \left( \frac{1}{n^2 \deg_o(u)} \right)$, because $(1 - 1/n)^{n - 1} \geq e^{-1}$. We have that the number of iterations until $a_u$ sees its optimal solution once is a geometric random variable with parameter $\Omega \left( \frac{1}{n^2 \deg_o(u)} \right)$; thus, its expectation is $O \left( n^2 \deg_o(u) \right)$. (See, for example, Motwani and Raghavan [5] for a review on probabilistic analysis.)

To process $u$, we need $a_u$ to see its optimal solutions for $O \left( \frac{1}{\rho} \log \left( \frac{\tau_{\min}}{\tau_{\max}} \right) \right) = O \left( \frac{1}{\rho} \log n \right)$ times. By linearity of expectation, the expected number of iterations to process $u$ is $O \left( \frac{1}{\rho} n^2 \deg_o(u) \log n \right)$.

We then analyze the total running time. Let $T$ be the number of iterations to process every node. Consider nodes in topological order where $t$ is the first node and if there is an edge $(u, v) \in E$, nodes $v$ comes before $u$ in the ordering. Let $T_u$, be the number of iterations needed to process node $u$ after all nodes preceding $u$ are processed. From the previous calculation, we have that $E[T_u] = O \left( \frac{1}{\rho} n^2 \deg_o(u) \log n \right)$. Since $T = \sum_{u \in V} T_u$, we get

$$E[T] = E \left[ \sum_{u \in V} T_u \right] = \sum_{u \in V} O \left( \frac{1}{\rho} n^2 \log n \deg_o(u) \right)$$
$$= O \left( \frac{1}{\rho} n^2 m \log n \right),$$

because $\sum_{u \in V} \deg_o(u) = m$. □

Using standard techniques, we can turn the bound into a high probability bound with a multiplicative factor of $O \left( \log n \right)$ to the running time.

3. DAG with degree bounds

When we have upper bounds $\Delta$ on the out-degree and $D$ on the number of edges on any shortest paths, we can modify the values of $\tau_{\min}$ and $\tau_{\max}$ to get a better running time bound. The main concern is to get a constant lowerbound on the probability $p$ that $a_u$ chooses all correct edges after the first edge. Previously we get the bound of $(1 - (1 - \Delta) \cdot \frac{1}{\Delta^n})^n > e^{-1}$.

With these parameters, we can improve the bound from Lemma 1 to $\Omega \left( \frac{1}{\Delta \deg_o(u)} \right)$. The running time can be improved to $O \left( \frac{1}{\rho} m \Delta D \log(\Delta D) \right)$. The following theorem states this bound.

**Theorem 4** Let $\Delta$ be the maximum out-degree of $G$ and $D$ be the maximum number of edges on any shortest paths to $t$. The expected running time of an ANT with $\tau_{\min} = 1/\Delta D$ and $\tau_{\max} = 1 - \tau_{\min}$ runs in expected time $O \left( \frac{1}{\rho} m \Delta D \log(\Delta D) \right)$.

3.1. A running time on One-Max

One-Max is a simple optimization problem where there are $n$ zero-one variables, $x_1, \ldots, x_n$, with an objective to maximize $\sum_{i=1}^{n} x_i$, i.e., One-Max aims to maximize the number of variables having value 1.

Neumann and Witt reduces One-Max with $n$ variables to the longest path problem on a DAG with $3n + 1$ nodes and $4n$ edges. First, we note that the longest path and shortest path problems are equivalent on a DAG. Furthermore, in their reduction, every node has out-degree at most 2; also, we have $D \leq n$ for any graphs. Thus, Theorem 4 implies a running time of $O \left( \frac{1}{\rho} n^2 \log n \right)$ as claimed.

4. Graphs with cycles

We note that our proof can easily be modified to give the upperbound on the expected number of iterations of the $n$-ANT when applied to general graphs. However, the time for each iteration might be exponential when the graph contain cycles. This is clear in the case of directed graphs, since there exists a family of graphs where random walks required exponential steps. For undirected graphs, it is not clear to us how to deal with the fact that some edge may have a very low probability of being chosen (e.g., $1/n$), so it seems that bounds on random walks in
simple graphs are not enough to show the expected time for each iteration.

References


