Traffic Signal Optimization Using Ant Colony Algorithm

David Renfrew, Xiao-Hua Yu

Abstract— **Traffic signal control is an effective way to improve the efficiency of traffic networks and reduce users' delays. Ant Colony Optimization (ACO) is a meta-heuristic algorithm based on the behavior of ant colonies searching for food. ACO has successfully been employed to solve many complicated combinatorial optimization problems and its stochastic and decentralized nature fits well with traffic networks. This research investigates the application of the ant colony algorithm to minimize user delay at traffic intersections. Various ACO algorithms are discussed and a rolling horizon approach is also employed to achieve real-time adaptive control. Computer simulation results show that this new approach outperforms conventional fully actuated control, especially under the condition of high traffic demand.**

I. Introduction

Traffic network is an integral part of the civil infrastructures in many major metropolitan cities. Traffic congestion causes excess vehicle delays, leading to various issues and concerns such as safety, air pollution, and energy consumption ([1]).

There are many difficulties that need to be addressed in traffic signal control. Traffic movements are generally stochastic and non-linear; thus many conventional control techniques cannot yield optimal results. Also, traffic conditions can change quickly; accordingly, the signal control strategies must be highly responsive in real-time. As traffic networks grow in size, finding the optimal strategy becomes a complex combinatorial problem. Thus, advanced techniques in control and optimization must be employed.

TRANSYT (Traffic Network Study Tools; [2]), SCOOT (Split, Cycle and Offset Optimization Technique; [3]), and SCATS (Sydney Coordinated Adaptive Traffic System; [4], [5], and [6]) are conventional on-line control strategies based on the off-line optimization techniques. Detectors monitor traffic flows and predict future arrivals by creating flow profiles. The control strategies are then selected from a set of pre-determined off-line timing plans that match the current traffic flow profiles.

OPAC (Optimized Policies for Adaptive Control; [7]) incorporates a rolling horizon approach for optimization. A long time interval (usually 60 seconds) is considered; and all possible signal cycles over this time interval are sequentially evaluated to find the best switching timing plan. However, this optimal control policy is only implemented over a short period (usually around 4 seconds); and the whole process is then repeated. The long time period of optimization guarantees the performance of the timing plan, while the short implementation time ensures the algorithm is responsive to the time-varying traffic dynamics.

More recent research has introduced artificial neural networks into traffic control ([8], [9]). The advantage of neural networks is that no assumption on an analytic model for traffic flow needs to be made. However, neural network training can take a long time and require a large amount of data. Other latest developments on traffic signal control in recent years include fuzzy logic ([10]), Petri nets ([11]), and Markov decision control ([12]). PSO (Particle Swarm Optimization) has also been applied to traffic signal control at intersections with light traffic demands and showed some improvements (about 8.7%) on vehicle delay time over fixed signal cycle controls ([18]).

Ant Colony Optimization (ACO) is a meta-heuristic approach for solving computationally hard combinatorial optimization (CO) problems ([13], [14], and [15]). Inspired by the behavior of the ants in real world, ant colony algorithm is a multi-agent system, in which each single agent is called an artificial ant. It is one of the most successful examples of swarm intelligent systems and has been applied to solve many different types of problems, including the classical traveling salesman problem, path planning and network routing.

In nature, when searching for food, real ants may wander randomly until they find food. As an ant returns to the colony with food, it deposits pheromone, a chemical used for communication. These pheromone trails guide other ants as they continue their search for food. As more pheromone is deposited, the ants' paths become less random and are biased toward the paths with higher pheromone concentration.

In the ant colony algorithm, artificial ants search the solution space probabilistically to create candidate solutions. These candidate solutions are then evaluated and updated,

based on the pheromone associated with each one of them. It should be noticed that over time, certain amount of pheromone concentration may evaporate. Finally, the one with the highest value of pheromone is considered to be the optimal solution of the problem.

In this research, a new approach to find the optimal signal timing plan for a traffic intersection is investigated using ACO algorithm. Traffic signal problem is a complex combinatorial optimization problem which fits the nature of ACO very well. Rolling horizon algorithm is also employed to achieve real-time adaptive control. Computer simulation results indicates that this new approach is more efficient than traditional fully actuated control, especially under the conditions of high, but not saturated, traffic demand.

The rest of the paper is organized as follows. In section 2, the traffic flow model and dynamic equations at a typical intersection are summarized. In section 3, various ACO algorithms are discussed, including the ant system algorithm, the elitist ant system algorithm, and the rank-based ant system algorithm. In section 4, the above algorithms are applied and tested by simulation to find the optimal signal settings at a traffic intersection to minimize the average vehicle delay time. For each algorithm, the convergence rates with different parameters (e.g., number of ants) are studied and compared. A heuristic local search mechanism with weighted pheromone levels is considered to improve the performance of the ACO algorithm. Section 5 concludes the paper and also gives directions for future works.

II. The Traffic Flow Model

Modeling traffic dynamics and optimizing the control signals are two interrelated problems. Consider a typical four-lagged isolated traffic intersection with four external approaches, as shown in Fig. 1. For the sake of simplicity, only through movements are considered. The traffic flows move along two directions (east/west or north/south, labeled as 1, 2, 3, and 4 in the figure) and thus only two sets of traffic control signals (green for east/west while red for north/south, and green for north/south while red for east/west) are considered. In traffic engineering, directions (1 - 4 in Fig. 1) are also called "movements".

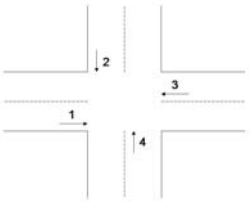

Fig. 1. A typical traffic intersection

At a given time *t*, the queue length on movement *i* can be denoted as $q^i(t)$, where *i* represents the index of a movement. Thus, the queue length at the whole intersection can be denoted as:

$$q(t) = \left[q^1(t), q^2(t), q^3(t), q^4(t)\right] \quad (1)$$

Similarly, the number of vehicles leaving movement *i* during a time interval (t_1, t_2) can be denoted as $q^i_{out}(t_1, t_2)$. It is a function of the signal choice and the queue length at t_1. When $u(t_1, t_2)$ = green, we have:

$$q^i_{out}(t_1, t_2) = min\left[q^i(t_1), 1 + Int\left(\frac{t_2 - t_1}{hw} \right) \right] \quad (2)$$

where *hw* is the headway between vehicles as they leave the intersection, $u(t_1, t_2)$ is the signal during the time interval (t_1, t_2) and $Int(\cdot)$ gives the integer part of the input. Obviously, when $u(t_1, t_2)$ = red, $q^i_{out}(t_1, t_2) = 0$.

The number of cars arriving during a time interval (t_1, t_2) can be denoted as $q_{in}(t_1, t_2)$. It has been supported by the results of many field tests that under most circumstances, the arrival of vehicles for the external movements follows the Poisson distribution [16]. Therefore,

$$P(n) = \frac{(\lambda \Delta t)^n e^{-\lambda \Delta t}}{n!} \quad (3)$$

where *n* is a positive integer for number of arrivals, λ is the average vehicle arrival rate in vehicles per hour and Δt is the duration of time period.

From the above, the dynamic equation of traffic flow can be described as:

$$q(t) = q(t-1) + q_{in}(t) - q_{out}(t) \quad (4)$$

III. The Ant Colony Algorithm

If The principle of swarm intelligence is based on the studies of social interactions between biological insects in nature. In contrast to the global, centralized traditional approach, it offers an alternative way to design an intelligent system based on the collective, decentralized behavior of many self-organized sub-systems.

A swarm intelligent system typically contains a population of simple agents which only interact locally with each other and the environment. That means, each individual agent in the system only follows simple rules and may not have the knowledge of the overall system. However, the local interactions between such agents can lead to the emergence of a very sophisticated and complicated group behavior. Some of the examples of biological swarm intelligent systems include ant colonies, bird flocking, fish schooling, bacterial growth, etc.

The ant colony optimization (ACO) algorithm was first developed by M. Dorigo in 1992 in his Ph. D. dissertation. It is a meta-heuristic approach for solving computationally hard combinatorial optimization (CO) problems; in other words, it is an “approximate” algorithm which can be used to obtain

"good enough solutions" in a reasonable amount of computation time ([13], [14], and [15]). Inspired by the foraging behavior of the biological ants in real world, artificial ants are introduced and employed as a novel computational intelligence tool. In fact, it is a stochastic search algorithm based on a parameterized probabilistic model called the pheromone model.

Consider a solution space in which each node represents a possible solution for an optimization problem. The major steps of ACO can be summarized as follows:

1) Initialization. The pheromone values on each node are set to a constant value.

2) Solution construction. Each ant begins on a start node and moves to one of its neighboring node based on the pheromone values. In general, ants move from node *i* to node *j* with the following probability (also called the proportional rule, or the transition probability):

$$p_{ij} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\ \eta_{ij}^{\beta}}{\sum_{l \in N_i} \tau_{il}^{\alpha}\ \eta_{il}^{\beta}} & if\ \ j \in N_i \\ 0 & if\ \ j \notin N_i \end{cases} \qquad (5)$$

where N_i is the set of the neighborhood nodes of *i* that an ant has not visited yet, which includes all possible nodes that an ant can move to when at node *i*. τ_{ij} is the pheromone value between node *i* and *j*; and η_{ij} represents the heuristic information (which is available a priori – for example, in the famous traveling salesman problem, the reciprocal of the distance between two different cities *i* and *j* is usually chosen to be η_{ij}). The values of α and β are usually application dependent; they weigh the importance of the pheromone and heuristic values, respectively. Note that there are potentially many different ways of choosing the transition probabilities; however, Eq. (5) was introduced in the first ACO algorithms, and is still used most often in ACO literature nowadays mainly due to this historical reason ([13]).

3) Update pheromone. Pheromone update can be implemented in different ways, depending on the specific algorithm being studied; but they all follow a general form. Over time, pheromone evaporates:

$$\tau_{ij}(n+1) = (1-\rho)\tau_{ij}(n) \qquad (6)$$

where *n* is the index of iteration; $\rho \in (0,1]$ is the evaporation rate. The pheromone on some of the paths is then updated by:

$$\tau_{ij}(n+1) = \tau_{ij}(n) + \Delta\tau_{ij} \qquad (7)$$

where $\Delta\tau_{ij}$, the pheromone update, is determined by the specific algorithm.

4) The above solution construction and pheromone update procedures (i.e., step 2 and 3) are repeated until a stop criterion is met.

ACO has been successfully applied to solve many different types of problems, including the classical traveling salesman problem, task assignment, path planning and routing in telecommunication network, etc. Many different ACO algorithms have been proposed, including the original Ant System algorithm (AS), Elitist Ant System, and Rank-based Ant System. In fact, the initialization and solution construction procedures are the same for different ACO algorithms; only the ways to update pheromone (i.e., step 3) are different. In this research, we consider three different ACO algorithms, namely, the Ant System (AS), the Elitist Ant System (EAS), and the Rank-based Ant System algorithm.

A. Ant System Algorithm:

In this algorithm, after all *m* ants have constructed their own solutions and the pheromones on all edges/arcs evaporate based on Eq. (6), the pheromones are updated by:

$$\tau_{ij}(n+1) = \tau_{ij}(n) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \qquad (8)$$

where $\Delta\tau_{ij}^{k}$, the pheromone deposited by ant *k* when moving from *i* to *j*, is defined by:

$$\Delta\tau_{ij}^{k} = \frac{1}{C^{k}} \qquad (9)$$

where C^{k} is the associated cost or reward. Otherwise (i.e., ant *k* doesn't move to node *j* from node *i*), there is no pheromone deposit, i.e., $\Delta\tau_{ij}^{k} = 0$.

B. Elitist Ant System Algorithm:

In Elitist Ant System (also called elitist strategy for ant system) algorithm, extra weight is given to the best-so-far solution. As in the Ant System algorithm, pheromone evaporates first (as in Eq. (6)), then is updated by:

$$\tau_{ij}(n+1) = \tau_{ij}(n) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e\Delta\tau_{ij}^{bs} \qquad (10)$$

where *e* is a weighting parameter. The additional term $\Delta\tau_{ij}^{bs}$ reinforces the best-so-far solution and can be defined as the following (if ant *k* moves from *i* to *j*):

$$\Delta\tau_{ij}^{bs} = \frac{1}{C^{bs}} \qquad (11)$$

where C^{bs} is the total cost/reward (from the start of the algorithm) associated with the best-so-far solution (including

the transition from *i* to *j*). This term can also be viewed as the pheromone deposited by an additional ant called the best-so-far ant.

C. Rank-Based Ant System:

In this algorithm, the ant's solutions are sorted in order of increasing cost before the pheromone is deposited. Only the *(w-1)* best-ranked ants and the best-so-far ant are allowed to deposit pheromone. The best-so-far solution is weighted by *w*; and the r^{th} best ant is weighted by *max{w-r, 0}*. Thus the pheromone update rule is:

$$\tau_{ij}(n+1) = \tau_{ij}(n) + \sum_{r=1}^{w-1}(w-r)\Delta\tau_{ij}^{r} + w\Delta\tau_{ij}^{bs} \tag{12}$$

where $\Delta\tau_{ij}^{r}$ and $\Delta\tau_{ij}^{bs}$ are defined in Eq. (9) and Eq. (11). The pheromone evaporation stage is performed before the update, as in the other methods, but less pheromone is generally evaporated on each step. The rank-based update biases away from bad solutions, allowing for more conservative evaporation.

IV. Computer Simulation Results

In this section, the ACO algorithm is applied to traffic signal optimization at a "four-legged" intersection as shown in Fig. 1. First, a simple case is considered to qualitatively study the performance of different ant colony algorithms. The convergence rates of pheromone concentration to the optimal solution using different algorithms and parameters are examined. Then, the proposed ACO algorithm is tested on a traffic intersection with various vehicle arrival rates, from 400 (vehicles per hour per movement) to 850 (vehicles per hour per movement); and the average vehicle delay of the ACO algorithm is compared with a traditional fully actuated control algorithm based on NEMA (National Electrical Manufacturers Association) standard. It is assumed that camera-type sensors are available at the intersection to monitor vehicle arrivals and/or departures. The traffic parameters used in simulations are summarized in Table 1.

Table 1. Traffic Simulation Parameters

Parameter	Value
Minimum green time (s)	5
Maximum green time (s)	30
All red time (s)	2
Minimum headway (s)	2
Extension time (s)	1

One of the most important goals of traffic signal control is to minimize vehicle waiting time at intersections. In this research, the amount of pheromone deposited by artificial ant is directly related with this performance criterion. As we know, the green time duration for each signal phase can be any value bounded between a minimum and a maximum value (called the minimum green and maximum green time). The inputs of the ACO controller include the current traffic queue (available from sensor measurements) and a prediction of vehicle waiting time; the output of ACO controller is the optimal signal switching time (or optimal time duration of the signal phase). It is assumed that the number of vehicles at the intersection is known, i.e., video-camera type detectors are available at the intersection. The ACO algorithm determines the optimal green time duration to minimize the total vehicle waiting time, which includes the actual waiting time of the vehicles already in the current queue, and the estimated waiting time of vehicles that may just arrive during this time duration ([17]):

Let's consider the situation when the length of a green signal is $(t_2 - t_1)$, where t_1 is the starting time, and $t_{min\_green} \le (t_2 - t_1) \le t_{max\_green}$. Let q be the queue length (number of vehicles) at time t_1, and $q \neq 0$.

Case 1. Green phase. When all vehicles in the initial queue are released, that is, when $(t_2 - t_1) \ge (q-1)\,hw$, the total expected waiting time for a traffic movement under green signal (from t_1 to t_2) can be written as:

$$J_{1green}(t_1, t_2) = \frac{q(q-1)hw}{2} + \sum_{i=1}^{q}(t_1 - t_a^i) + \frac{\lambda((q-1)\,hw)^2}{2} + \frac{\lambda((q-1)\,hw)[\lambda((q-1)\,hw)-1]hw}{2} \tag{13}$$

where t_a^i is the arrival time of vehicle *i*. The first and second terms are the total waiting time of the initial queue, the third term is the expected waiting time of vehicles that arrive during the time interval (t_1, t_2) when the initial vehicles are released, and the fourth term is the expected time that takes to release these new arrivals in (t_1, t_2).

Case 2. Green phase. When $(t_2 - t_1) < (q-1)\,hw$, not all the vehicles in the initial queue can be released. The total expected waiting time for this case is:

$$J_{2green}(t_1, t_2) = \frac{q_{out}(q_{out}-1)hw}{2} + \sum_{i=1}^{q}(t_1 - t_a^i) + \sum_{i=1}^{q}(t_1 - t_a^i) + (q - q_{out})(t_2 - t_1) + \frac{\lambda(t_2 - t_1)^2}{2} \tag{14}$$

The first term is the waiting time of the released vehicles in (t_1, t_2), the second term is the waiting time of the initial queue before t_1, the third term is the waiting time of the initial vehicles not being released in (t_1, t_2), and the fourth term is the expected waiting time of estimated arrivals in (t_1, t_2).

Case 3. Red phase. During the red phase, no vehicle can be released; in addition, $\lambda(t_2 - t_1)$ vehicles are expected to arrive. Therefore, the total queue at t_2 becomes $q + \lambda(t_2 - t_1)$. The expected total waiting time is:

$$J_{red}(t_1, t_2) = q(t_2 - t_1) + \sum_{i=1}^{q}(t_1 - t_a^i) + \frac{\lambda(t_2 - t_1)^2}{2} \quad (15)$$

A detailed discussion on the above equations (Eq. (13) - (15)) can be found in [17].

The number of ants used in ACO is an important implementation issue. As little as one ant could be used, but this does not take full advantage of the algorithm. When more ants are used, more explorations can be done during each iteration. As a result, more pheromone is released per iteration, decreasing the chance of biasing towards poor solutions. But, increasing the number of ants increases the computational work done per iteration. Additionally, the large amount of pheromone deposited does not allow significant bias towards the optimal path. As a result, the pheromone levels may change slowly and thus makes it more difficult to find the optimal solution. In this research, we compare the performance of each algorithm with 10, 25, and 50 ants.

To study the convergence rates of pheromone concentration to the optimal solution of different ant colony algorithms, a simple case of just one signal cycle is considered first. It is assumed that each movement of the intersection initially has zero vehicles in their queue, and the vehicle arrival rate is 800 vehicle/hour/movement. In this case, it is optimal to switch the signal after the minimum green time. For each choice of each ant colony algorithms and/or parameters, 100 trials are run. Fig. 2, 3, 4 and 5 show the average results of these 100 trials, with y-axis representing the percentage of the total pheromone deposited on the optimal signal (path) and x-axis representing the index of iterations.

Fig. 2 shows the average rate of convergence of the Ant System algorithm. The best result is obtained by using only 10 ants; however, the maximum percentage of the total pheromone on the optimal signal is only about 11%. The performance is the Elitist Ant System algorithm is illustrated in Fig. 3, where the maximum percentage of the total pheromone deposited on the optimal signal is increased to about 45% with 50 ants, showing a significant improvement over the Ant System algorithm.

One problem with the ACO is its tendency to accumulate pheromone on near optimal solutions ([13]). At initialization, all paths are chosen with equal probability. If a near optimal solution is chosen by some ants at the beginning, the positive feedback of the ant colony algorithm can cause pheromone to accumulate rapidly on this near optimal solution. As a result, the optimal path may not be found. When using an ACO algorithm with the best-so-far ant, this stagnation became especially apparent. To avoid stagnation, a search of the solutions near the best-so-far solution can be added. It is accomplished by replacing every the n^{th} iteration of the random solution search with a local search. In this local search the search space is replaced with a neighborhood of size T of the best-so-far solution. In this simulation, local search is performed every 3 iterations with the neighborhood size $T = 4$. The simulation result of the Elitist Ant System algorithm with local search is shown in Fig. 4, where all the pheromones are deposited on the optimal signal (for 10 ants) in its steady state. That is, the optimal solution is found after 60 iterations.

To achieve a faster rate of convergence, ants' solutions are ranked and the solutions with the highest cost function are discarded. Once the optimal solution is found pheromone accumulates more rapidly, because the best solutions are weighted heavier and the bad solutions are ignored. Comparing with other ACO algorithms, the rank-based ant system usually requires the fewest ants for similar performance, which is more computational efficient and thus more suitable for real-time applications.

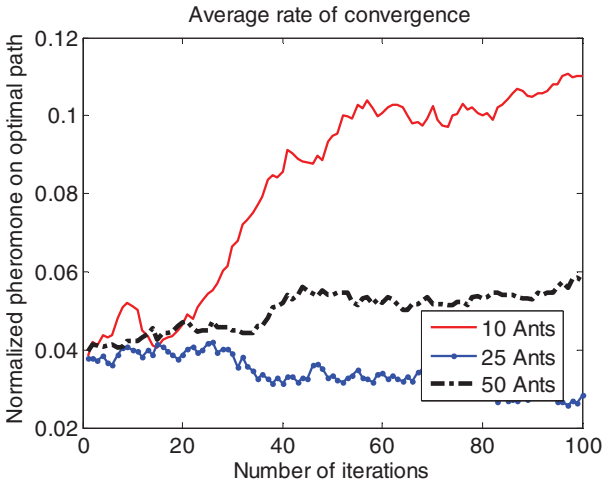

Fig. 2. The average rate of convergence of the Ant System algorithm

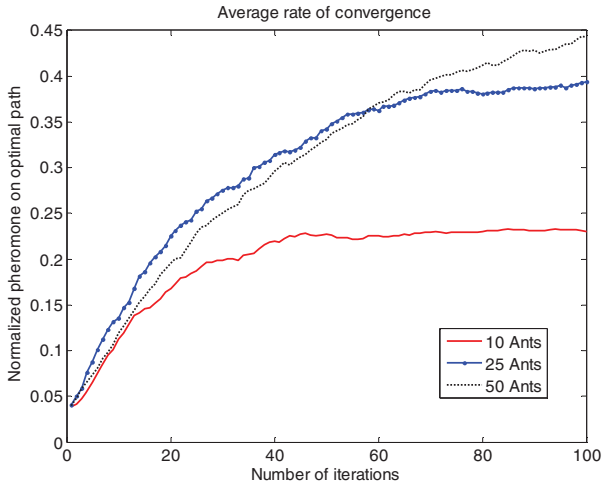

Fig. 3. The average rate of convergence of the Elitist Ant System algorithm

An advantage of the ACO is its ability to incorporate heuristic information about the solution space being searched ([13]). In the traffic signal problem, releasing all vehicles in the queue usually results in smaller waiting times; thus the green phase length should be set accordingly. For shorter queues,

releasing the current queue and then switching the signal is optimal. For longer queues, additional time is needed because additional vehicles may arrive before all vehicles (that are currently in the queue) are released. In either case, intuitively, the optimal green signal length is *around* the time that is needed to release all vehicles that are currently in the queue. This time interval is $t_1 + \left(q^g(t_1) - 1\right)hw$, where $q^g(t_1)$ is the length of the largest queue on the green movements at time t_1. To bias the search towards switch timing *near* this time, the pheromone levels in Eq. (5) can be weighted by the heuristic value of

$$\eta_{t_1 t_2} = \exp\left[\frac{\left(q^g(t_1) - 1\right)hw - (t_2 - t_1)}{c}\right] \qquad (16)$$

where c is a positive constant. The exponential function is chosen because it can provide a maximum value at the desired peak location and smooth transitions on both sides of the peak. The performance of the rank-based ant system with local search and heuristics is demonstrated in Fig. 5. Obviously, it yields the fastest convergence rate (comparing with other ACO algorithms such as the ant system and the elitist ant system).

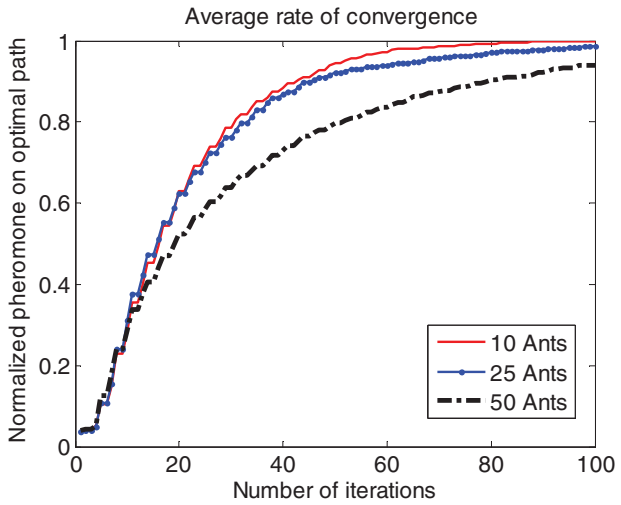

Fig. 4. The average rate of convergence of the Elitist Ant System algorithm with local search

The proposed ACO algorithm (rank-based ant system with local search and heuristic) is tested and compared with the conventional fully actuated control by computer simulation. In fully actuated control algorithm, both the cycle length and the green time for every phase of the intersection can be varied. At every time step, the fully actuated controller checks whether an arrival has occurred on any lane of the intersection. If an arrival has occurred, then the phase is given an extension if it has a green indication. If the phase does not have a green, a call is registered for that phase. To determine the signal indication of next phase, all the calls need to be taken into account. The phase sequence of fully actuated control is fixed; however, certain phases in the cycle may be skipped if there is no demand detected by detectors.

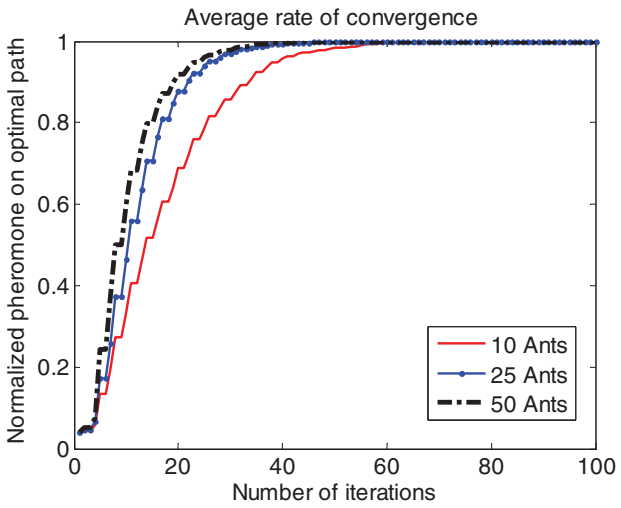

Fig. 5. The average rate of convergence of the Rank-based Ant System with local search and heuristics

It is assumed that the intersection is "clear" when the simulation starts (i.e., zero initial conditions, or no queue at the beginning), and each traffic movement is independent. The traffic simulation runs for ten minutes, allowing traffic flow to "settle" and also to reduce the effects of initial conditions; then vehicle delays are recorded and compared (ACO vs. the traditional fully actuated control). A vehicle's delay is defined as the time difference between its departure time and arrival time. The average delay (per vehicle) is defined as:

$$Average\_delay = \frac{\sum_{i=1}^{N}(t_d^i - t_a^i)}{N} \qquad (17)$$

where t_d^i and t_a^i is the departure time and arrival time of the i^{th} vehicle, respectively. The average is taken over the all N vehicles that arrive during the time period of consideration. As stated in section 2, it is a general assumption that the vehicle arrival pattern follows Poisson distribution ([16]); therefore, 40 different random sets of data are simulated for each arrival rate. The time resolution of all the simulations is 0.01 seconds.

When vehicle arrival rate is low (i.e., light traffic), the fully actuated controller performs better. For example, when the arrival rate is less than 600 vehicles per hour per movement, the expected vehicle inter-arrival time (i.e., the time between two adjacent arrivals) is greater than 6 seconds. The minimum green time is 5 seconds and with the 2 second all red time; so the minimum time between a phase transition is 7 seconds. The probability of a vehicle arriving during a red signal is small (though cannot be ignored). The fully actuated controller is better suited for this situation - the signal changes only when there is a vehicle that arrives during red signal.

Once traffic flow rate is greater than 600 vehicles per hour per movement, the number of vehicles that arrive per red signal is frequently greater than one. The fully actuated controller gives too much preference to the green direction; while the proposed ACO algorithm takes all movements into account and performs better. Fig. 6 plots the average vehicle delay for both

fully actuated control and ACO algorithm over 40 trials. As traffic approaches saturation, the average delay for the fully actuated control increases much faster than in ACO control. For example, when the vehicle arrival rate is 850 (vehicles/hour/movement), the ACO algorithm shows about 85% improvement over the fully actuated control in terms of the average delay time.

The ant colony simulations run fast enough to be effectively implemented in real time systems. For example, it only takes about eight minutes (on a PC) to simulate twenty minutes of traffic flows and controls (with the rank-based ant system using local search, elitist ant, heuristic weights and ten ants).

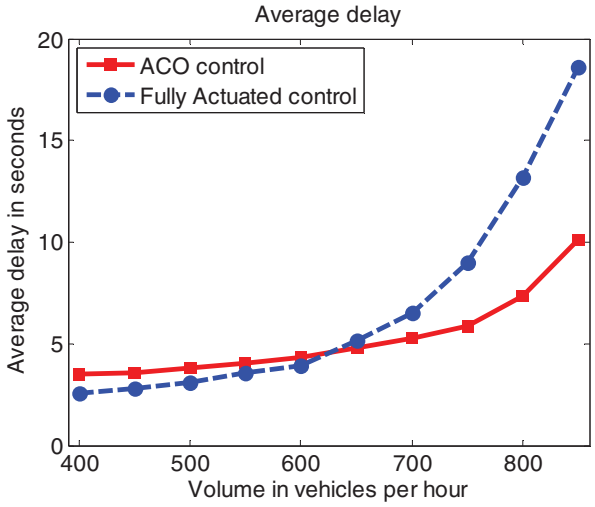

Fig. 6. The average delay

V. Conclusion

ACO (Ant Colony) algorithm is a new optimization technique based on swarm intelligence. In this paper, the rank-based ant system algorithm with local search and heuristic is applied to control signals at traffic intersection to reduce the vehicle waiting time. Computer simulation results show this method outperforms the conventional fully actuated control under the situation of high traffic demand. In addition, the ant colony algorithms are fast enough to be effectively implemented in real time systems. Further evaluation and testing on this approach will be performed.

References

[1] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and W. Yibing, "Review of road traffic control strategies," Proceedings of the IEEE, vol. 91, pp. 2043-2067, 2003.

[2] Transportation research center, Traffic network study tool: TRANSYT-7F software summary, University of Florida, 1987

[3] P. Hunt, and D. Robertson, "The SCOOT on-line traffic signal optimization technique", *Traffic engineering and control*, April 1982.

[4] J. Y. K. Luk, "Two traffic-responsive area traffic control methods: SCAT and SCOOT," Traffic engineering and control, 1984.

[5] P. Hunt, and D. Robertson, "The SCOOT on-line traffic signal optimization technique", *Traffic engineering and control*, April 1982.

[6] P. Lowrie, "The Sydney coordinated adaptive control systems – principles, methodology, algorithms", IEE conference publication, vol. 207, 1982.

[7] N. Gartner, "OPAC: A demand-responsive strategy for traffic signal control," Transportation research record, no. 906, 1983.

[8] S. Chien, Y. Ding, "Dynamic Bus Arrival Time Prediction with Artificial Neural Networks," *J. Trans. Engrg*., vol. 128, 2002.

[9] M. Papageorgiou, A. Messmer, J. Azema, and D. Drewanz, "A neural network approach to freeway network traffic control," *Control Engineering Practice,* vol. 3, pp. 1719-1726, 1995.

[10] W. Wei, et. al., "Traffic signal control using fuzzy logic and MOGA," *Proceedings of IEEE conference on systems, man, and cybernetics, 2001*.

[11] G. List, and M. Cetin, "Modeling traffic signal control using Petri nets", IEEE Transaction on intelligent transportation systems, vol. 5, no. 3, 2004.

[12] X.-H. Yu, and A. Stubberud, "Markovian decision control for traffic signal systems", *Proceedings of the 36th IEEE conference on decision and control*, 1997.

[13] M. Dorigo, and T. Stutzle, *Ant Colony optimization*, The MIT Press, 2004.

[14] M. Dorigo, and C. Blum, "Ant colony optimization theory: A survey", Theoretical Computer Science, vol. 344, 2005, pp. 243 – 278.

[15] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization - Artificial ants as a computational intelligence technique", IEEE Computational Intelligence magazine, vol. 1, issue 4, 2006, pp. 28-39.

[16] R. Wilshire, R. Black, et al., *Traffic control systems handbook*, FHWA-IP-85-12, 1985.

[17] D. Renfrew, and X.-H. Yu, "Traffic Signal Control with Swarm Intelligence", Proceedings of the International Conference on Natural Computation, pp. 79 – 83, August 2009.

[18] L. Zhang; Y. Zhong; Z. Li and Y. Chen, "PSO-based optimization for isolated intersections signal timings and simulation", Proceedings of the International Conference on Machine Learning and Cybernetics, pp. 993 – 996, 2008.