Parallelization Strategies for the Ant System

Lorenzo Cioni

Dipartimento di Informatica

e-mail: lcioni@di.unipi.it
Topics of my talk

- A few words on *metaheuristics*
- A few words on *parallel processing*
- The *Ant System* (serial version)
- The *Ant System* (parallel versions)
- Exploiting Parallelism
- Algorithmic issues
- Conclusions
Metaheuristics are strategies that “guide” the search process, their goal is to efficiently explore the search space so to find (if any) optimal solutions.

Metaheuristics range from simple local search procedures to complex learning processes.

Metaheuristics are not problem specific and usually make use of domain specific knowledge in the form of heuristics.

Metaheuristics make use of (well balanced) diversification (move to unexplored areas of the search space) and intensification (intensively explore areas of the search space) techniques.
Classification of Metaheuristics

- Nature-inspired vs. non-nature-inspired
  - Genetic Algorithms, Ant Algorithms
  - Tabu Search, Iterated Local Search

- Population-based vs. Single Point Search or Trajectory Methods
  - Genetic Algorithms, Ant Algorithms
  - Tabu Search, Iterated Local Search, Variable Neighborhood Search

- Dynamic vs. static objective function
  - o.f. modified during search or kept unchanged

- One vs. various neighborhood structures

- Memory usage vs. memory-less methods
  - Use or not of the search history, short-long term memory
“True” parallel computing (MIMD): concurrent execution of control flows on data flows

Two approaches:
- shared memory: concurrent access to memory locations, conflicts
- message passing: communication overhead

Three models:
- Synchronous: synchronization points (fork-join), communication overhead
- Asynchronous: independent flows, local minima
- Partially Asynchronous: mixed approach, ratio local computation/global computation
Parallel Computing 2

- **Parameters:**
  - the ratio of computation, communication and idle times in relation to the total simulated execution time
  - the speedup $S(N) = T(1)/T(N)$, $N$ = number of processors
  - the efficiency $E(N) = S(N)/N$
  - the efficacy $\eta = S(N)E(N)$

- **Exploiting parallelism (to be refined):**
  - Analytical techniques
  - Simulation models
  - Measurement experiments
Ant System 1

- Metaheuristic
  - nature-inspired, population-based
    - real ants (population) searching for food

- Basic elements:
  - cooperating agents (artificial ants)
  - set of rules:
    - generation
    - update
    - usage
      - of local and global information so to find good solutions
  - local heuristic function: examination of feasible solutions
  - artificial ants searching the solution space mimic real ants looking for food
Ant System 2
Traveling Salesperson Problem

- Complete weighted graph $G = (V, E, d)$, $V = \{v_i : i=1, .., n\}$, $E = \{(v_i, v_j) : i \neq j\}$, $d_{ij}$ weight (distance or cost) of the arc $(v_i, v_j)$;
- minimum cost hamiltonian tour;
- given $n$ cities TSP:
  - the $m$ artificial ants are distributed on the $n$ cities according to some rule;
  - at the start of each iteration all cities but the assigned ones can be visited ($\Omega$);
  - each ant decides independently which (not yet visited) city to visit next (Tabu list);
- selection probability of $j$ from $i$ ($p_{ij}$) varies directly with the pheromone trail (intensity, adaptive memory, parameter $\alpha$) and inversely with distance (visibility, parameter $\beta$);
- the city selection process is repeated until all ants have completed a tour;
- at each step of an iteration $\Omega = \Omega \setminus \{j\}$, if $\Omega = \{k\}$ then $k$ with $p_{ik} = 1$;
- each ant $k$ evaluates the length of the tour $L_k$: a best tour is found and updated;
- the trail levels of pheromone are updated (every ant has the same quantity per tour);
- the shorter the tour the more pheromone per unit length;
- (analogy to nature) pheromone evaporation ($\rho$): avoids early convergence.
Ant System 3
Traveling Salesperson Problem: probability and pheromone update

\( p_{ij} = \begin{cases} \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{n \in \Omega} [\tau_{in}]^{\alpha}[\eta_{in}]^{\beta}} & \text{if } j \in \Omega \\ 0 & \text{otherwise} \end{cases} \)

where \( \eta_{ij} = \frac{1}{d_{ij}} \)

\( \tau_{ij} \) intensity of trail between cities \( v_i \) and \( v_j \)
\( \alpha \) parameter to regulate the influence of \( \tau_{ij} \)
\( \eta_{ij} \) visibility of city \( v_j \) from city \( i \)
\( \beta \) parameter to regulate the influence of \( \eta_{ij} \)
\( \Omega \) set of cities, that have not been visited yet
\( d_{ij} \) distance between cities \( v_i \) and \( j \)

\( \tau_{ij}(t+1) = \rho \tau_{ij}(t) + \Delta \tau_{ij} \) \( (2) \)

where \( \Delta \tau_{ij} = \sum_{k=1}^{m} \Delta \tau_{ij}^k \) and \( \Delta \tau_{ij}^k = \begin{cases} \frac{1}{L_k} & \text{if ant } k \text{ travels on edge } (v_i, v_j) \\ 0 & \text{otherwise} \end{cases} \)

where \( t \) iteration counter
\( \rho \in [0, 1] \) parameter to regulate the reduction of \( \tau_{ij} \)
\( \Delta \tau_{ij} \) total change of trail level on edge \( (v_i, v_j) \)
\( m \) number of ants
\( \Delta \tau_{ij}^k \) change of trail level on edge \( (v_i, v_j) \) caused by ant \( k \)
\( L_k \) length of tour found by ant \( k \)
Ant System: the sequential version

Initialize
For $t = 1$ to $T$
    For $k = 1$ to $m$ do
        Repeat until ant $k$ has completed a tour
            Select city $v_j$ to be visited next
            with probability $p_{ij}$ given by equation (1)
        Calculate the length $L_k$ of the tour generated by ant $k$
        Update the trail levels $\tau_{ij}$ on all edges according to equation (2)
    End
End

- $T$ iterations, $n$ cities $m$ ants: $O(Tmn^2)$
- $m=n$, one ant in each city: $O(m^3)$
- "Natural" parallelism: during each iteration ants behave independently from each other
**Ant System: parallelization**

- **Synchronous (left)**
  - fork
  - send $D$
  - send $\tau_0$
  - compute tour, $L_k$
  - send tour, $L_k$
  - update, check
  - send $\tau_{i,j}$
  - compute tour, $L_k$
  - ...

- **Partially asynchronous (right)**
  - fork
  - send $D$
  - send $\tau_0$
  - compute tour, $L_k$
  - send tour, $L_k$
  - local update, check
  - send tour, $L_k$
  - global update, check

---

**TSP tours in parallel**

- one ant one process

- one worker one group of ants
Parallel Ant System: speedup

**Basic hypotheses (a little bit unreal):**

- no communication overhead, infinite number of processing elements (workers), 1 process (ant) -1 worker

\[ S_{\text{asymptotic}}(m) = \frac{T_{\text{seq}}(m)}{T_{\text{par}}(m, \infty)} = \frac{\mathcal{O}(m^3)}{\mathcal{O}(m^2)} = \mathcal{O}(m) \]

**More realistic assumptions:**

- communication overhead, finite number of workers \( N \ll m \) (number of ants), 1 set of processes -1 worker (load balancing)

\[ S(m, N) = \frac{\mathcal{O}(m^3)}{\mathcal{O}(m^3/N) + T_{\text{ovh}}(m, N)} \]

**Partially asynchronous solution:**

- 1 set of processes -1 worker, local iterations and global synchronization
- reduced communication overhead, good values may be “broadly” ignored
- ratio local/global is a crucial parameter (5 in the experiments)
Exploiting parallelism

- **Behavior evaluation:**
  - *analytical techniques*
    - abstract, simplified model of parallel program characteristics, complexity in estimating communication overhead
  - *simulation models*
    - discrete event simulation
    - input: description of the parallel program structure (three computational tasks: compute tour, local update, global update, two communication blocks: broadcast of trails, collection of paths)
    - input: resource requirement specification
    - assumption: time to send a message = fixed startup + variable time depending on the size of the message
    - assumption: multiple simultaneous communications without contention
    - output: trace file with time stamps of starts and stops of each task/block
  - measurement experiments on a real implementation
    - data dependent
Synchronous vs. Partially Asynchronous 1
Synchronous vs. Partially Asynchronous

1. the ratio of computation, communication and idle times in relation to the total simulated execution time,
2. the speedup $S(N) = T(1)/T(N)$,
3. the efficiency $E(N) = S(N)/N$, and
4. the efficacy $\eta(N) = S(N) \cdot E(N)$. 
Variants

- Gain in speedup with the same quality or better quality with the same speed or both;
- Synchronous: rule for grouping processes and assigning to workers;
- Partially Asynchronous: also ratio local/global iterations $I_i$, $i=1, ..., N$;
  - the higher $I_i$ the lower the communication overheads but the easier workers get trapped in local minima:
    - static approach: $I_i$ constant;
    - dynamic approach: $I_i$ from low to high;
- Processes (or ants) grouping:
  - assignment to workers: random or rule based assignment (distance criterion or quality)
  - dynamics: assignment only once or repeated after several global or local computations;
- Ants ranking according to solution quality so that only best ranked ants can update trails
- Use of local search to improve the solution generated by artificial ants
Closing remarks

- two parallelization strategies
  - synchronous (S),
  - partially asynchronous (PA, local/global = 5 in the experiments)
- discrete event simulation to evaluate performances
- (PA) performs better than (S) owing to reduced communication frequency among workers (very important on real parallel architectures)
Bibliographical references


- Bernd Bullnheimer, Gabriele Kotsis, Christine Strauß, “Parallelization Strategies for the Ant System”, Report Series, Report n° 8, October 1997, University of Economics and Business Administration, Vienna