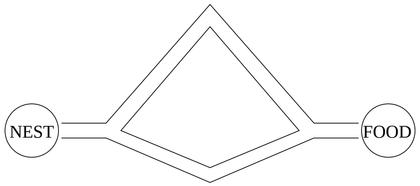
11 Novel Methods

- Ant Algorithms
- Message Passing Methods

11.1 Ant Algorithms

- Dorigo et al. (1991 onwards), Hoos & Stützle (1997), ...
- Inspired by experiment of real ants selecting the shorter of two paths (Goss et al. 1989):



- Method: each ant leaves a *pheromone trail* along its path; ants make probabilistic choice of path biased by the amount of pheromone on the ground; ants travel faster along the shorter path, hence it gets a differential advantage on the amount of pheromone deposited.

Ant Colony Optimisation (ACO)

- Formulate given optimisation task as a path finding problem from source s to some set of valid destinations $t_1, \ldots, t_n$ (cf. the A^* algorithm).
- Have agents ("ants") search (in serial or parallel) for candidate paths, where local choices among edges leading from node i to neighbours $j \in N_i$ are made probabilistically according to the local "pheromone distribution" τ_{ij}:

$$p_{ij} = \frac{\tau_{ij}}{\sum_{j \in N_i} \tau_{ij}}.$$

- After an agent has found a complete path π from s to one of the t_k, "reward" it by an amount of pheromone proportional to the quality of the path, $\triangle\tau \propto q(\pi)$.

- Have each agent distribute its pheromone reward $\triangle\tau$ among edges (i,j) on its path π: either as $\tau_{ij} \leftarrow \tau_{ij} + \triangle\tau$ or as $\tau_{ij} \leftarrow \tau_{ij} + \triangle\tau/\text{len}(\pi)$.
- Between two iterations of the algorithm, have the pheromone levels "evaporate" at a constant rate $(1-\rho)$:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij}.$$

ACO motivation

- Local choices leading to several good global results get reinforced by pheromone accumulation.
- Evaporation of pheromone maintains diversity of search. (I.e. hopefully prevents it getting stuck at bad local minima.)
- Good aspects of the method: can be distributed; adapts automatically to online changes in the quality function $q(\pi)$.
- Good results claimed for Travelling Salesman Problem, Quadratic Assignment, Vehicle Routing, Adaptive Network Routing etc.

ACO variants

Several modifications proposed in the literature:

- To exploit best solutions, allow only best agent of each iteration to distribute pheromone.
- To maintain diversity, set lower and upper limits on the edge pheromone levels.
- To speed up discovery of good paths, run some local optimisation algorithm on the paths found by the agents.
- Etc.

An ACO algorithm for the TSP (1/2)

- Dorigo et al. (1991)
- At the start of each iteration, m ants are positioned at random start cities.
- Each ant constructs probabilistically a Hamiltonian tour π on the graph, biased by the existing pheromone levels. (NB. the ants need to remember and exclude the cities they have visited during the search.)
- In most variations of the algorithm, the tours π are still locally optimised using e.g. the Lin-Kernighan 3-opt procedure.
- The pheromone award for a tour π of length $d(\pi)$ is $\triangle\tau = 1/d(\pi)$, and this is added to each edge of the tour: $\tau_{ij} \leftarrow \tau_{ij} + 1/d(\pi)$.

An ACO algorithm for the TSP (2/2)

- The local choice of moving from city i to city j is biased according to weights:

$$a_{ij} = \frac{\tau_{ij}^{\alpha}(1/d_{ij})^{\beta}}{\sum_{j \in N_i} \tau_{ij}^{\alpha}(1/d_{ij})^{\beta}},$$

  where $\alpha, \beta \geq 0$ are parameters controlling the balance between the current strength of the pheromone trail τ_{ij} vs. the actual intercity distance d_{ij}.
- Thus, the local choice distribution at city i is:

$$p_{ij} = \frac{a_{ij}}{\sum_{j \in N_i'} a_{ij}},$$

  where N_i' is the set of permissible neighbours of i after cities visited earlier in the tour have been excluded.

11.2 Message Passing Methods

Belief Propagation (or the *Sum-Product Algorithm*):

- Pearl (1986) and Lauritzen & Spiegelhalter (1986).
- Originally developed for probabilistic inference in graphical models; specifically for computing marginal distributions of free variables conditioned on determined ones.
- Recently generalised to many other applications by Kschischang et al. (2001) and others.
- Unifies many other, independently developed important algorithms: Expectation-Maximisation (statistics), Viterbi and “Turbo” decoding (coding theory), Kalman filters (signal processing), etc.
- Presently of great interest as a search heuristic in constraint satisfaction.

Survey Propagation

- Braunstein, Mézard & Zecchina (2005).
- Refinement of Belief Propagation to dealing with “clustered” solution spaces.
- Based on statistical mechanics ideas of the structure of configuration spaces near a “critical point”.
- Remarkable success in solving very large “hard” randomly generated Satisfiability instances.
- Success on structured problem instances not so clear.

Belief propagation

- Method is applicable to any constraint satisfaction problem, but for simplicity let us focus on Satisfiability.
- Consider cnf formula F determined by variables $x_1,\dots,x_n$ and clauses $C_1,\dots,C_m$. Represent truth values as $\xi \in \{0,1\}$.
- Denote the set of satisfying truth assignments for F as

$$\mathcal{S} = \{x \in \{0,1\}^n \mid C_1(x) = \cdots = C_m(x) = 1\}.$$

- We aim to estimate for each variable x_i and truth value $\xi \in \{0,1\}$ the bias of x_i towards ξ in $\mathcal{S}$:

$$\beta_i(\xi) = \Pr_{x\in\mathcal{S}}(x_i = \xi).$$

- If for some x_i and ξ, $\beta_i(\xi) \approx 1$, then x_i is a “backbone” variable for the solution space, i.e. most solutions $x \in \mathcal{S}$ share the feature that $x_i = \xi$.

Bias-guided search

If the biases β_i could be computed effectively, they could be used e.g. as a heuristic to guide backtrack search:

function BPSearch(F: cnf):
  if F has no free variables **then return** val$(F) \in \{0,1\}$
  else
    $\bar{\beta} \leftarrow$ BPSurvey(F);
    choose variable x_i for which $\beta_i(\xi) =$ max;
    val $\leftarrow$ BPSearch($F[x_i \leftarrow \xi]$);
    if val $= 1$ **then return** 1
    else return BPSearch($F[x_i \leftarrow (1-\xi)]$);
  end if.

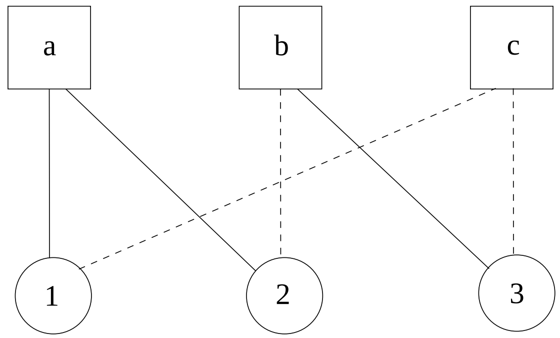
Alternately, the bias values could be used to determine variable flip probabilities in some local search method etc.

Message passing on factor graphs

- The problem of course is that the biases are in general difficult to compute. (It is already NP-complete to determine whether $\mathcal{S} \neq \emptyset$ in the first place.)
- Thus, the BP survey algorithm aims at just estimating the biases by iterated local computations ("message passing") on the factor graph structure determined by formula F.
- The factor graph of F is a bipartite graph with nodes $1, 2, \ldots$ corresponding to the variables and nodes $a, b, \ldots$ corresponding to the clauses. An edge connects nodes i and u if and only if variable x_i occurs in clause C_u (either as a positive or a negative literal).

A factor graph

Factor graph representation of formula $F = (x_1 \vee x_2) \wedge (\bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_3)$:



Belief messages

- The BP survey algorithm works by iteratively exchanging "belief messages" between interconnected variable and clause nodes.
- The variable-to-clause messages $\mu_{i \to a}(\xi)$ represent the "belief" (approximate probability) that variable x_i would have value ξ in a satisfying assignment, *if* it was not influenced by clause C_a.
- The clause-to-variable messages $\mu_{a \to i}(\xi)$ represent the belief that clause C_a can be satisfied, *if* variable x_i is assigned value ξ.

Propagation rules

- Initially, all the variable-to-clause messages are initialised to $\mu_{i \to a}(\xi) = 1/2$.
- Then beliefs are propagated in the network according to the following update rules, until no more changes occur (a fixpoint of the equations is reached):

$$\mu_{i \to a}(\xi) = \frac{\prod_{b \in N_i \setminus a} \mu_{b \to i}(\xi)}{\prod_{b \in N_i \setminus a} \mu_{b \to i}(\xi) + \prod_{b \in N_i \setminus a} \mu_{b \to i}(1-\xi)}$$

$$\mu_{a \to i}(\xi) = \sum_{x : x_i = \xi} C_a(x) \cdot \prod_{j \in N_a \setminus i} \mu_{j \to a}(x_j)$$

  (Here notation $N_u \setminus v$ means the neighbourhood of node u, excluding node v.)
- Eventually the variable biases are estimated as $\beta_i(\xi) \approx \mu_{i \to a}(\xi)$.

Belief propagation: limitations (1/2)

- The belief update rules entail strong independence assumptions about the variables. E.g. in the update rule for $\mu_{a\to i}(\xi)$ it is assumed that the probability $\Pr_{x\in\mathcal{S}}(x_j=\xi_j,\, j\in N_a\setminus i)$ factorises as $\prod_{j\in N_a\setminus i}\mu_{j\to a}(x_j)$. Thus the estimated variable biases may not be the correct ones.
- Furthermore, the message propagation may never converge to stable message values. However it is known that if the factor graph is a tree (contains no loops), then a stable state is reached in a single two-way pass from leaf variable nodes to a chosen root node and back.

T–79.4201 Search Problems and Algorithms

Belief propagation: limitations (2/2)

- Even if the correct bias values $\beta_i(\xi)=\Pr_{x\in\mathcal{S}}(x_i=\xi)$ were known, these may be noninformative in the case when the solution space is “clustered”.
- For instance, assume there are cn, $c>0$, “backbone” variables whose different assignments lead to different types of solution families. Then it may be the case that all $\beta_i\approx 1/2$ also for these variables, even though for any solution cluster they are in fact highly constrained.
- The more advanced Survey Propagation algorithm aims to address this problem.

I.N. & P.O. Autumn 2007