

12 Complexity of Search

- ▶ The “No Free Lunch” Theorem
- ▶ Combinatorial Phase Transitions
- ▶ Complexity of Local Search

The NFL theorem: definitions (1/4)

- ▶ Consider family \mathcal{F} of all possible objective functions mapping finite search space \mathcal{X} to finite value space \mathcal{Y} .
- ▶ A **sample** d from the search space is an ordered sequence of distinct points from \mathcal{X} , together with some associated cost values from \mathcal{Y} :

$$d = \{(d^x(1), d^y(1)), \dots, (d^x(m), d^y(m))\}.$$

Here m is the **size** of the sample. A sample of size m is also denoted by d_m , and its projections to just the x - and y -values by d_m^x and d_m^y , respectively.

- ▶ The set of all samples of size m is thus $\mathcal{D}_m = (\mathcal{X} \times \mathcal{Y})^m$, and the set of all samples of arbitrary size is $\mathcal{D} = \cup_m \mathcal{D}_m$.

12.1 The “No Free Lunch” Theorem

- ▶ Wolpert & Macready 1997
- ▶ Basic content: All optimisation methods are equally good, when averaged over uniform distribution of objective functions.
- ▶ Alternative view: Any nontrivial optimisation method *must* be based on assumptions about the space of relevant objective functions. [However this is very difficult to make explicit and hardly any results in this direction exist.]
- ▶ Corollary: one cannot say, unqualified, that ACO methods are “better” than GA’s, or that Simulated Annealing is “better” than simple Iterated Local Search. [Moreover as of now there are *no* results characterising some nontrivial class of functions \mathcal{F} on which some interesting method \mathcal{A} would have an advantage over, say, random sampling of the search space.]

The NFL theorem: definitions (2/4)

- ▶ An **algorithm** is any function a mapping samples to *new* points in the search space. Thus:

$$a : \mathcal{D} \rightarrow \mathcal{X}, \quad a(d) \notin d^x.$$

- ▶ *Note 1:* The assumption $a(d) \notin d^x$ is made to simplify the performance comparison of algorithms; i.e. one only takes into account *distinct* function evaluations. Not all algorithms naturally adhere to this constraint (e.g. SA, ILS), but without it analysis is difficult.
- ▶ *Note 2:* The algorithm may in general be stochastic, i.e. a given sample $d \in \mathcal{D}$ may determine only a *distribution* over the points $x \in \mathcal{X} - d^x$.

The NFL theorem: definitions (3/4)

- ▶ A **performance measure** is any mapping Φ from cost value sequences to real numbers (e.g. minimum, maximum, average). Thus:

$$\Phi : \mathcal{Y}^* \rightarrow \mathbb{R},$$

where $\mathcal{Y}^* = \cup_m \mathcal{Y}^m$:



The NFL theorem: statement

Theorem

[NFL] For any value sequence d_m^y and any two algorithms a_1 and a_2 :

$$\sum_{f \in \mathcal{F}} P(d_m^y | f, m, a_1) = \sum_{f \in \mathcal{F}} P(d_m^y | f, m, a_2).$$



The NFL theorem: definitions (4/4)

- ▶ Finally, denote by $P(d_m^y | f, m, a)$ the probability distribution of value samples of size m obtained by using a (generally stochastic) algorithm a to sample a (typically unknown) function $f \in \mathcal{F}$.
- ▶ More precisely, such a sample is obtained by starting from some a -dependent search point $d^x(1)$, querying f for the value $d^y(1) = f(d^x(1))$, using a to determine search point $d^x(2)$ based on $(d^x(1), d^y(1))$, etc., up to search point $d^x(m)$ and the associated value $d^y(m) = f(d^x(m))$. The value sample d_m^y is then obtained by projecting the full sample d_m to just the y -coordinates.



The NFL theorem: corollaries

Corollary

[1] Assume the uniform distribution of functions over \mathcal{F} , $P(f) = 1/|\mathcal{F}| = |\mathcal{Y}|^{-|X|}$. Then for any value sequence $d_m^y \in \mathcal{Y}^m$ and any two algorithms a_1 and a_2 :

$$P(d_m^y | m, a_1) = P(d_m^y | m, a_2).$$

Corollary

[2] Assume the uniform distribution of functions over \mathcal{F} . Then the expected value of any performance measure Φ over value samples of size m ,

$$E(\Phi(d_m^y) | m, a) = \sum_{d_m^y \in \mathcal{Y}^m} \Phi(d_m^y) P(d_m^y | m, a),$$

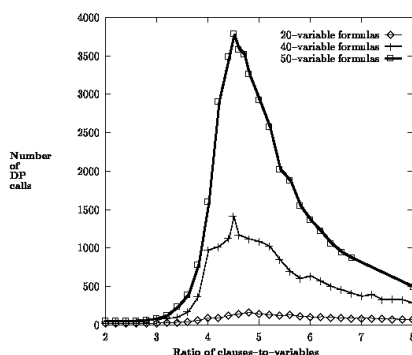
is independent of the algorithm a used.



12.2 Combinatorial Phase Transitions

- ▶ “Where the Really Hard Problems Are” (Cheeseman et al. 1991)
- ▶ Many NP-complete problems can be solved in polynomial time “on average” or “with high probability” for reasonable-looking distributions of problem instances. E.g. Satisfiability in time $O(n^2)$ (Goldberg et al. 1982), Graph Colouring in time $O(n^2)$ (Grimmett & McDiarmid 1975, Turner 1984).
- ▶ Where, then, are the (presumably) exponentially hard instances of these problems located? Could one tell ahead of time whether a given instance is likely to be hard?
- ▶ Early studies: Yu & Anderson (1985), Hubermann & Hogg (1987), Cheeseman, Kanefsky & Taylor (1991), Mitchell, Selman & Levesque (1992), Kirkpatrick & Selman (1994), etc.

Hard instances for 3-SAT (2/4)



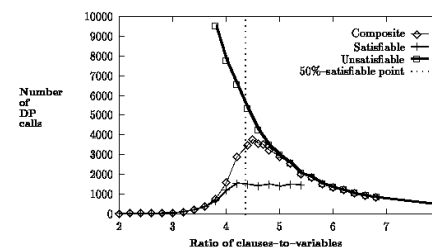
Results:

- ▶ A distinct peak in median running times at about clauses-to-variables ratio $\alpha \approx 4.5$.
- ▶ Peak gets more pronounced for increasing $n \Rightarrow$ well-defined “delta” distribution for infinite n ?

Hard instances for 3-SAT (1/4)

- ▶ Mitchell, Selman & Levesque, AAAI-92
- ▶ Experiments on the behaviour of the DPLL procedure on randomly generated 3-cnf Boolean formulas.
- ▶ Distribution of test formulas:
 - ▶ n = number of variables
 - ▶ $m = \alpha n$ randomly generated clauses of 3 literals, $2 \leq \alpha \leq 8$
- ▶ For sets of 500 formulas with $n = 20/40/50$ and various α , Mitchell et al. plotted the median number of recursive DPLL calls required for solution.

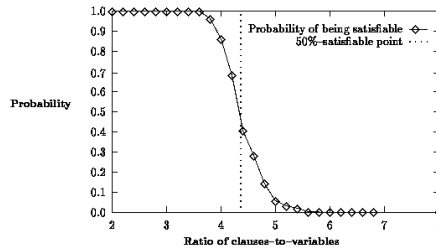
Hard instances for 3-SAT (3/4)



- ▶ The runtime peak seems to be located near the point where 50% of formulas are satisfiable.
- ▶ The peak seems to be caused by relatively short unsatisfiable formulas.

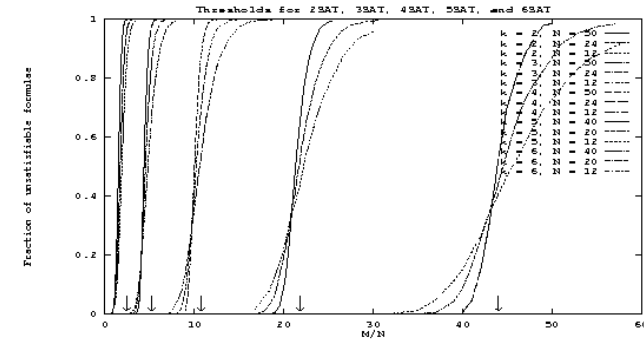
Question: Is the connection of the running time peak and the satisfiability threshold a characteristic of the DPLL algorithm, or a (more or less) algorithm independent “universal” feature?

The satisfiability transition (1/2)



Mitchell et al. (1992): The “50% satisfiable” point or “satisfiability threshold” for 3-SAT seems to be located at $\alpha \approx 4.25$ for large n .

The satisfiability transition (2/2)



Kirkpatrick & Selman (1994):

- ▶ Similar experiments as above for k -SAT, $k = 2, \dots, 6$, 10000 formulas per data point.
- ▶ The “satisfiability threshold” α_c shifts quickly to larger values of α for increasing k .

Statistical mechanics of k -SAT (1/4)

Kirkpatrick & Selman, *Science* 1994

A “spin glass” model of a k -cnf formula:

- ▶ variables $x_i \sim$ spins with states ± 1
- ▶ clauses $c \sim k$ -wise interactions between spins
- ▶ truth assignment $\sigma \sim$ state of spin system
- ▶ Hamiltonian $H(\sigma) \sim$ number of clauses unsatisfied by σ
- ▶ $\alpha_c \sim$ critical “interaction density” point for “phase transition” from “satisfiable phase” to “unsatisfiable phase”

Statistical mechanics of k -SAT (2/4)

Estimates of α_c for various values of k via “annealing approximation”, “replica theory”, and observation:

k	α_{ann}	α_{rep}	α_{obs}
2	2.41	1.38	1.0
3	5.19	4.25	4.17 ± 0.03
4	10.74	9.58	9.75 ± 0.05
5	21.83	20.6	20.9 ± 0.1
6	44.01	42.8	43.2 ± 0.2

Statistical mechanics of k -SAT (3/4)

The “annealing approximation” means simply assuming that the different clauses are satisfied independently. This leads to the following estimate:

- ▶ Probability that given clause c is satisfied by random σ :
 $p_k = 1 - 2^{-k}$.
- ▶ Probability that random σ satisfies all $m = \alpha n$ clauses assuming independence: $p_k^{\alpha n}$.
- ▶ Total number of satisfying assignments $= 2^n p_k^{\alpha n} \triangleq S_k^n(\alpha)$.
- ▶ For large n , $S_k^n(\alpha)$ falls rapidly from 2^n to 0 near a critical value $\alpha = \alpha_c$. Where is α_c ?
- ▶ One approach: solve for $S_k^n(\alpha) = 1$.

$$S_k^n(\alpha) = 1 \Leftrightarrow 2p_k^\alpha = 1$$

$$\Leftrightarrow \alpha = -\frac{1}{\log_2 p_k} = -\frac{\ln 2}{\ln(1 - 2^{-k})} \approx \frac{\ln 2}{2^{-k}} = (\ln 2) \cdot 2^k$$

◀ ▶ ⏪ ⏩ 🔍

Statistical mechanics of k -SAT (4/4)

It is in fact known that:

- ▶ A sharp satisfiability threshold α_c exists for all $k \geq 2$ (Friedgut 1999).
- ▶ For $k = 2$, $\alpha_c = 1$ (Goerdts 1982, Chvátal & Reed 1982). Note that 2-SAT \in P.
- ▶ For $k = 3$, $3.145 < \alpha_c < 4.506$ (lower bound due to Achlioptas 2000, upper bound to Dubois et al. 1999).
- ▶ Current best empirical estimate for $k = 3$: $\alpha_c \approx 4.267$ (Braunstein et al. 2002).
- ▶ For large k , $\alpha_c \sim (\ln 2) \cdot 2^k$ (Achlioptas & Moore 2002).

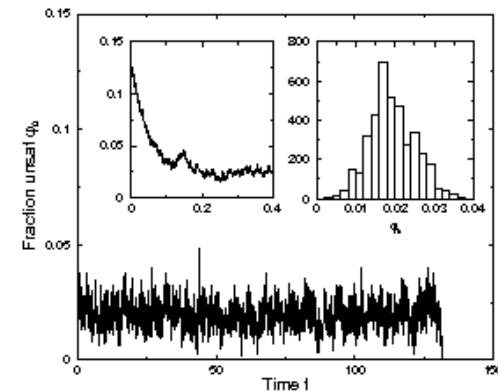
◀ ▶ ⏪ ⏩ 🔍

12.3 Complexity of Local Search

- ▶ Good experiences for 3-SAT in the satisfiable region $\alpha < \alpha_c$: e.g. GSAT (Selman et al. 1992), WalkSAT (Selman et al. 1996).
- ▶ *Focusing* the search on unsatisfied clauses seems to be an important technique: in the (unsystematic) experiments in Selman et al. (1996), WalkSAT (focused) outperforms NoisyGSAT (unfocused) by several orders of magnitude.

◀ ▶ ⏪ ⏩ 🔍

Dynamics of local search



A WalkSAT run with $p = 1$ (“focused random walk”) on a randomly generated 3-SAT instance, $\alpha = 3$, $n = 500$: evolution in the fraction of unsatisfied clauses (Semerjian & Monasson 2003).

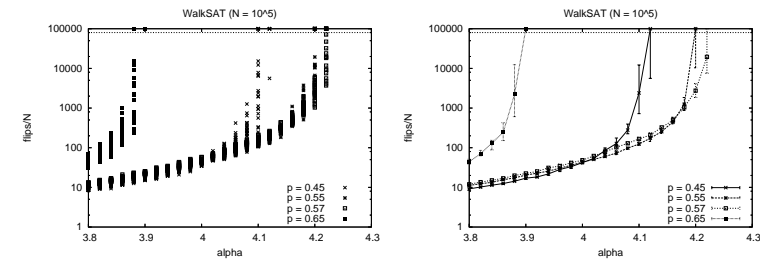
◀ ▶ ⏪ ⏩ 🔍

Some recent results and conjectures

- ▶ Barthel, Hartmann & Weigt (2003), Semerjian & Monasson (2003): WalkSAT with $p = 1$ has a “dynamical phase transition” at $\alpha_{\text{dyn}} \approx 2.7 - 2.8$. When $\alpha < \alpha_{\text{dyn}}$, satisfying assignments are found in linear time per variable (i.e. in a total of cn “flips”), when $\alpha > \alpha_{\text{dyn}}$ exponential time is required.
- ▶ Explanation: for $\alpha > \alpha_{\text{dyn}}$ the search equilibrates at a nonzero energy level, and can only escape to a ground state through a large enough random fluctuation.
- ▶ Conjecture: all local search algorithms will have difficulties beyond the so called “clustering transition” at $\alpha \approx 3.92 - 3.93$ (Mézard, Monasson, Weigt et al.)

Some WalkSAT experiments

For $p > 1$, the α_{dyn} barrier for linear solution times can be broken (Aurell & Kirkpatrick 2004; Seitz, Alava & Orponen 2005).

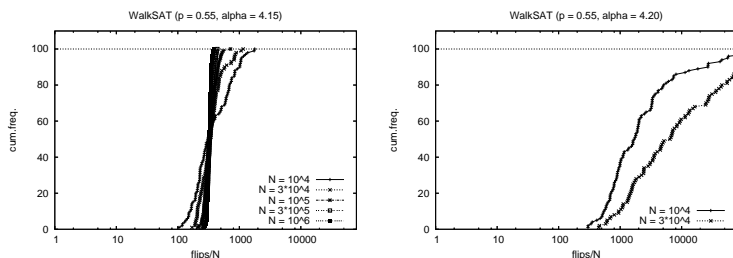


Normalised (flips/ n) solution times for finding satisfying assignments using WalkSAT, $\alpha = 3.8 \dots 4.3$.

Left: complete data; right: medians and quartiles.

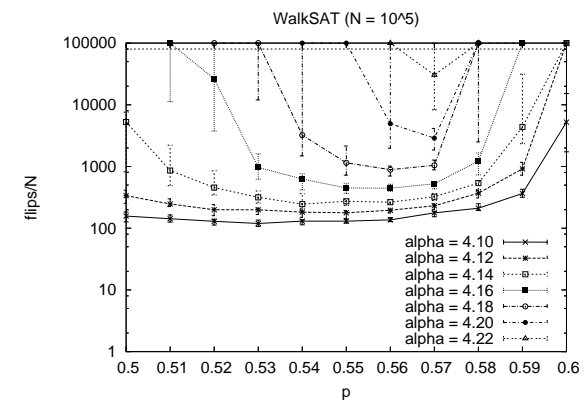
Data suggest linear solution times for $\alpha \gg \alpha_{\text{dyn}} \approx 2.7$.

WalkSAT linear scaling



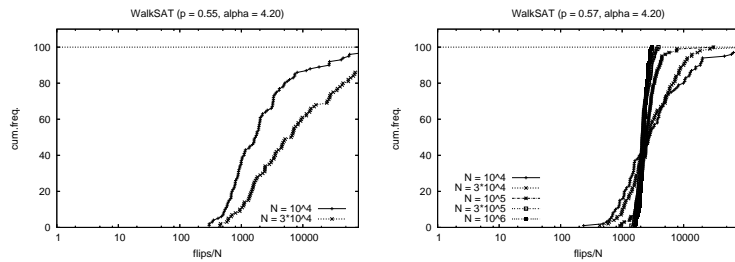
Cumulative solution time distributions for WalkSAT with $p = 0.55$.

WalkSAT optimal noise level?



Normalised solution times for WalkSAT with $p = 0.50 \dots 0.60$, $\alpha = 4.10 \dots 4.22$.

WalkSAT sensitivity to noise

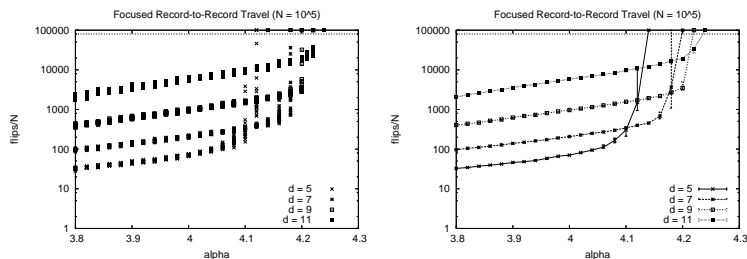


Cumulative solution time distributions for WalkSAT at $\alpha = 4.20$ with $p = 0.55$ and $p = 0.57$.

RRT applied to random 3-SAT

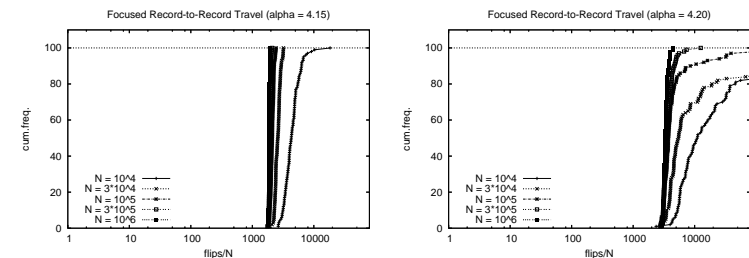
- ▶ Similar results as for WalkSAT are obtained with the Record-to-Record Travel algorithm.
- ▶ In applying RRT to SAT, $E(s)$ = number of clauses unsatisfied by truth assignment s . Single-variable flip neighbourhoods.
- ▶ Focusing: flipped variables chosen from unsatisfied clauses. (Precisely: one unsatisfied clause is chosen at random, and from there a variable at random.) \Rightarrow FRRT = focused RRT.

FRRT experiments (3-SAT)



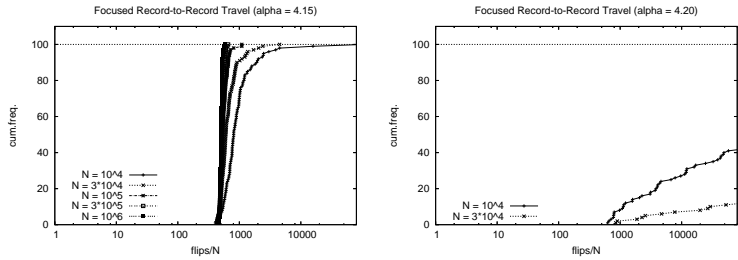
Normalised solution times for FRRT, $\alpha = 3.8 \dots 4.3$.
Left: complete data; right: medians and quartiles.

FRRT linear scaling (1/2)



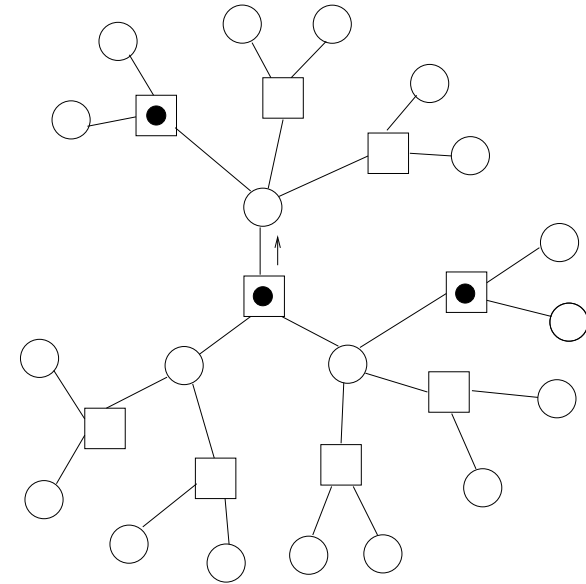
Cumulative solution time distributions for FRRT with $d = 9$.

FRRT linear scaling (2/2)

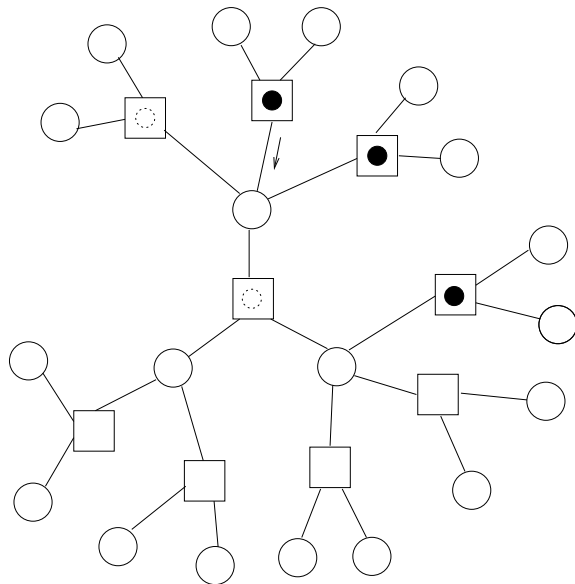


Cumulative solution time distributions for FRRT with $d = 7$.

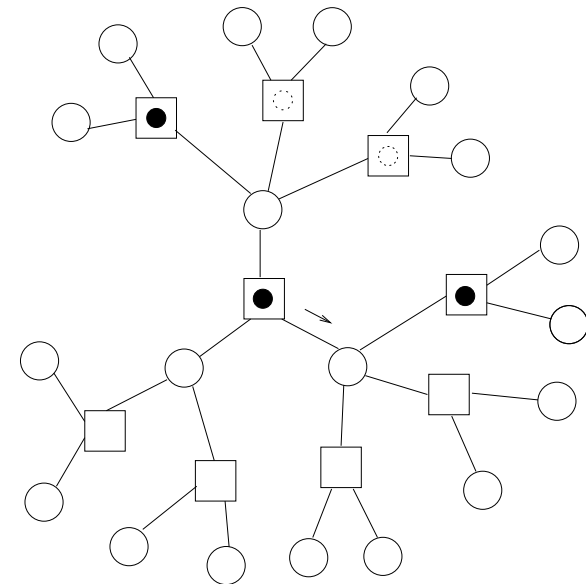
Focused search as a contact process



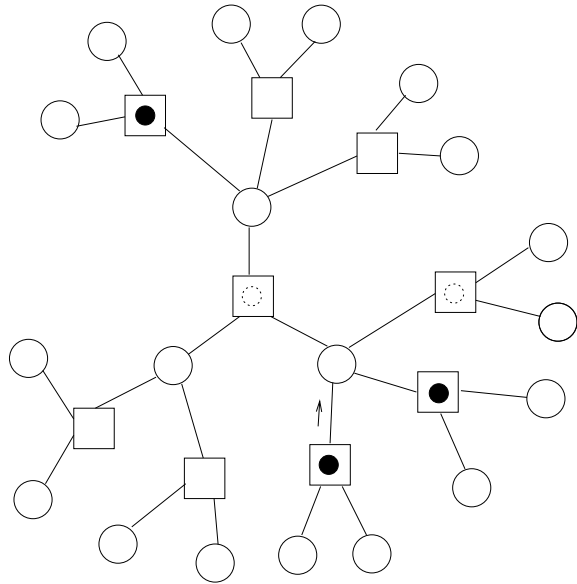
Focused search as a contact process



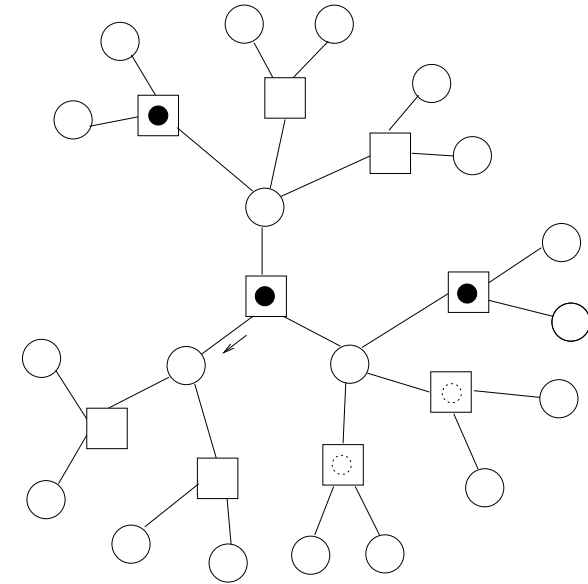
Focused search as a contact process



Focused search as a contact process



Focused search as a contact process



Focused search as a contact process

