An introduction to CarThaGène

An integrated generic/radiated hybrid/comparative mapping tool

T. Schiex
S. de Givry, M. Bouchez, P. Chabrier, C. Gaspin

INRA (Toulouse), France

February 11, 2006

Genetic and RH maps

Given a dataset (`dsload`) of genetic/RH data on a set of markers M, a genetic/RH map is defined by:

- a set of markers $N = \{m_1, \dots m_n\} \subseteq M$
- which is ordered (eg. $m_1 < \dots < m_n$)
- with a distance between each pair of adjacent markers ($d(m_i, m_{i+1})$)

The genetic/RH mapping problem: find a map (order+distances) that best explains the data set.

What is a good map ?

- Non parametric approach: minimizes the number of compulsory crossovers/breaks, maximizes the sum of 2-points LOD...
- Parametric approach: maximizes the probability of the data (likelihood) under a probabilistic model.

  Parameters: probability of recombination/breakage between 2 adj. markers θ_{ij} (probability of retention r)

CarthaGène criteria: multipoint maximum likelihood. May use non parametric approaches to guide the search.

Probabilistic models in CarthaGène

1. Backcross: as in MapMaker. Dedicated EM.
2. RIL (sib/self): as in MapMaker. Dedicated EM.
3. F2 Intercross: as in MapMaker.
4. Phase known outbreds (1:1, 1:2:1, 1:1:1:1 seg. ratio)
5. Haploid RH: Dedicated EM.
6. Diploid RH.

"Dedicated EM": can run more than 2 orders of magnitude faster than existing EM implementation (MapMaker, RHMAP).

Working with multiple populations

1. Consensus mapping (`dsmergen`): one map for all populations.

   All populations share the same marker ordering and distances. Can be used only for similar population types (eg. backcross with outbreds. RIL sib/self merge with RIL sib/self resp. only).

2. Simultaneous mapping (`dsmergor`): one order for all populations.

   No assumption that distances are the same. Can be used to merge eg. genetic and RH data, or RIL with BC.

Consensus mapping

Based on EM: Expectation/Maximization. Untyped in a population = missing (ignored by 2 point measures).

Expectation

For every adjacent markers, need to compute the expected number of cross-overs givent the data and a recombination probability θ. All expectations are pooled (multiple pop.)

Maximization

Computes a new $\hat{\theta}$ estimate based on new expectations. Analytical optimization (RIL+BC = 4^{th} degree eq.).

Simultaneous mapping

Under a given order:

LogLikelihood computation

is performed for each population indepedently using the order on defined markers.

LogLikelihood combination

All loglikelihood are added together to compute the criteria.

- Simpler: one loglikelihood computation per population.
- But: one consensus order and population specific recombination ratios.

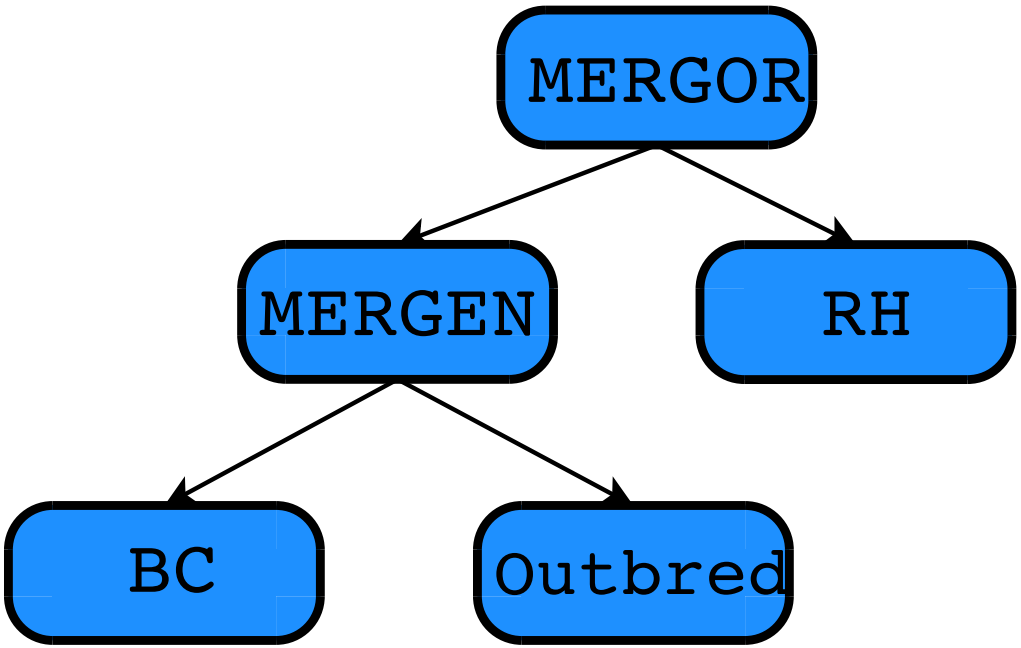
Taking into account extra information in CarthaGène

- Penalized likelihood: uses a priori information on supposedly "true" marker ordering. `constraints` dataset type.

  $\langle m_i, m_j, m_k \rangle, p$

  Maps such that m_j is not in between m_i and m_k have loglikelihood penalized by p. Usable with `dsmergor`.
- Comparative mapping: uses a known order on a related organism (eg. dog/human). Based on a breakpoint based bayesian model. Uses `dsmergor`.

Can be combined



Computing linkage groups

As in MapMaker: given a distance threshold θ_{max} and a LOD threshold ℓ_{min}, pool markers that have:

1. a pairwise distance below θ_{max}
2. a LOD above ℓ_{min}

Weakness: 2 unrelated markers can be pooled just if they are enough related to one marker (`group`, `groupget`).

2-points information is computed on loading (`dsload`, `mrklod2p`, `mrkfr2p`).

Ordering markers and the TSP

In a group of n markers, there are $\frac{n!}{2}$ differents orders.

Under strong hypothesis (BC, no missing), maximum likelihood ordering is equivalent to the

Wandering Salesman Problem

Given n cities and inter-cities distances, find a path that passes once through each city and that minimizes the overall distance.

One of the most studied optimization problem in computer science. Know to be potentially very hard (NP-hard).

Ordering markers: use TSP link

- exhaustive search: not possible for $n > 8$.
- building heuristics: simple map construction guided by multi or 2-point data (or user provided) (`sem`, `mfmapl/d`, `nicemapd/l`, `buildfw...`).
- improving heuristics: improve existing maps using a simple systematic mechanism (`flips`, `polish`)
- stochastic search: improve an existing map by complex stochastic perturbations (`greedy`, `annealing`, `lkh*...`)

All "improving" methods can be used as map checking methods (good = cannot be improved).

Good maps & the Heap

Good maps

Not only max. likelihood maps. May be also reliable map (no alternative order has comparable likelihood).

For the same set of "active markers" (`mrkselset`, `mrkselget`), CarThaGène remembers the k best maps encountered by the ordering procedures.

The heap

The set of the k best maps found. Gives a representation of the neighborhood of the current optimum map (`heaprints`, `heaprintds`, `heapget`...).

Mining the heap

`heaprinto` *n comp blank*

For each map in the heap, compares the sequence of the markers with the best sequence.

- if $n > 0$: only markers that moved are visualized and contiguous segments of more than n markers that moved are put in brackets.
- if *comp* is set, the output is unaligned. This is useful when the maps include a large number of markers.
- if *blank* is set, the segments which have been moved and whose length is $> n$ are only represented by their extremities.

Assess map reliable/unreliable regions/markers.

Heuristics building method

Initial map building

- by hand: specify a marker ordering and ask for max. likelihood estimation. `markselset` + `sem` : single EM.
- WSP heuristics:
  - `nicemapd/l`: using "Nearest Neighbor" heuristics (distances or LOD)
  - `mfmap/l`: uses "Multi Fragment" heuristics.

Warning: 2-points distances/LOD may be undefined/null when merging data-sets. Try all and keep best multipoint.

Framework building method

`buildfw` Δ_{min} Δ_{keep} S c $(S = \{\}, c = 0)$

1. Start from all possible pairs of markers.
2. For all available maps, a new marker is inserted in all possible positions. The marker "reliability" is defined as the difference δ in loglike between the best and the second best insertion position. A marker can be inserted only if this difference is larger than Δ_{min}.
3. From all these new maps, keep only those such that $\delta \geq \Delta_{keep}$.
4. repeat to 2.

From framework to comprehensive

`buildfw` Δ_{min} Δ_{keep} S c

- S: a marker ordering to start from (rather than all pairs). Used to extend an existing "reliable" map.
- $c = 1$: when no marker with sufficient quality exists, tries to independently insert all remaining markers in all possible intervals.
  For each such marker: reports the best insertion position (+) and how far in loglike all other positions are (support for the best position).

Heuristics improving methods

Start from a non empty heap (best taken).

`flips` w Δ_{max} *Iter*

Tests all maps obtained by permutations inside a sliding window of size w.
Reports all permutations with loglike. within Δ_{max} of the best known loglike (reliability assessment).
If an improved map is found and if *Iter* is set to 1, the process is reiterated on the new best map.

`polish`

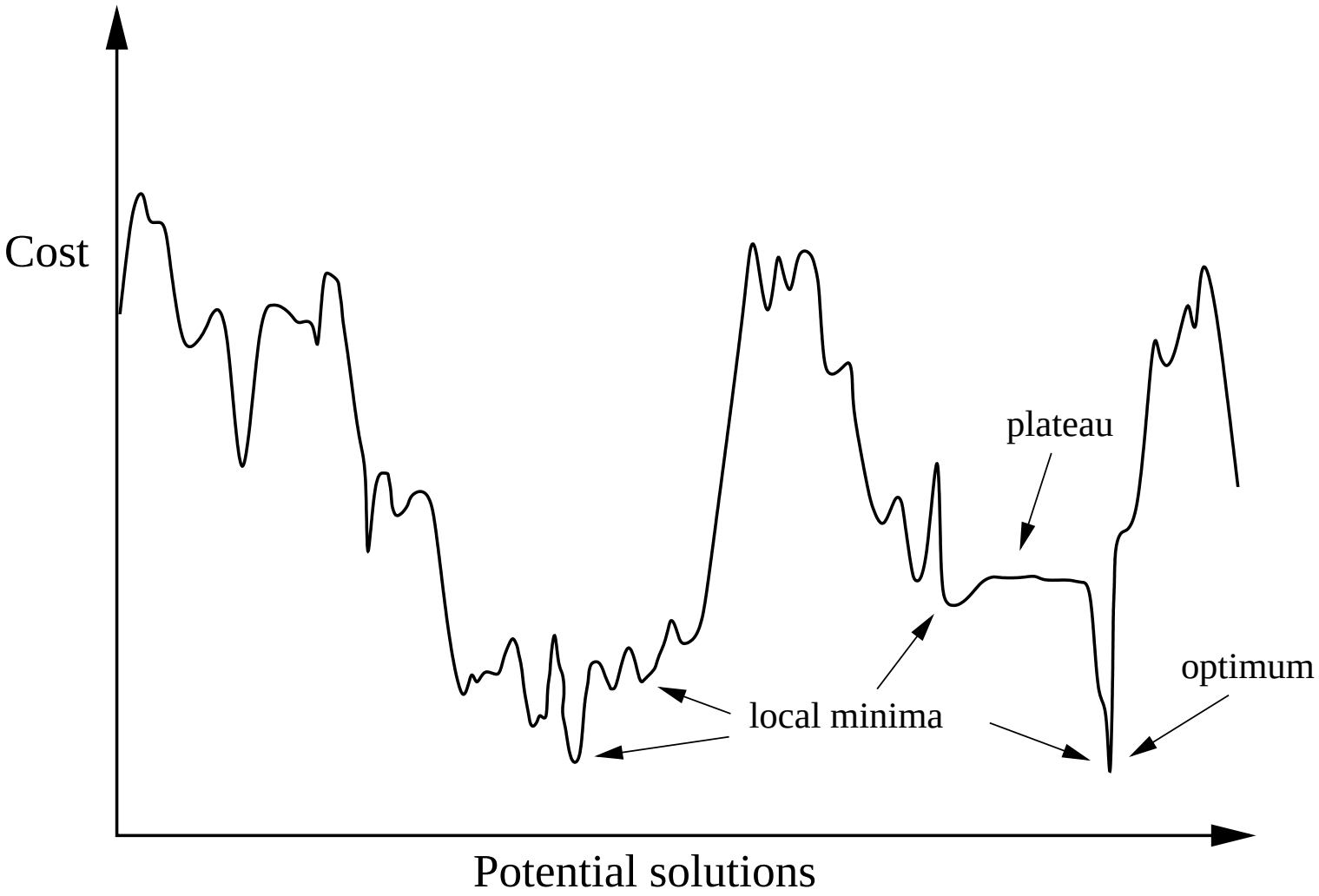
Each marker in the map is tested in all possible intervals. Reports the matrix of the Δ in loglike.

Stochastic search: general principles

1. We start from the best order available in the heap.
2. We perturbate this order to get a new order (called a neighbor). The neighbor chosen may be chosen randomly or "smartly". The loglike may increase or decrease.
3. depending on some tests (which may include stochasticity) we either "move" to this new order or stay were we are.
4. we repeat from 2.

The whole process may be repeated several time.
The possible perturbations (the neighborhood) is crucial: use known TSP neighborhood.

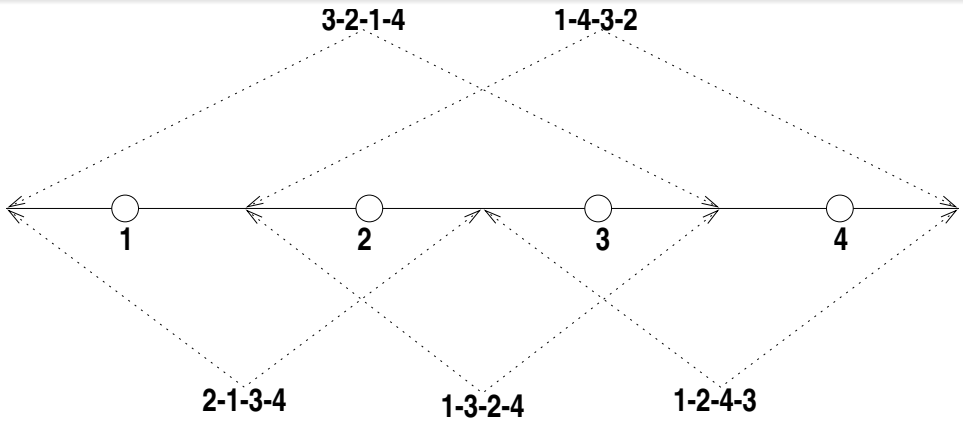
Stochastic search: general principles



Neighborhood

2-OPT, 3-OPT and Lin-Kernighan (WSP)

- 2-OPT choose 2 markers and invert the delimited submap.
- 3-OPT: choose 3 markers and swap the two delimited submaps.
- LKH: complex neighborhood (involving several k-OPT). Non GPL code, very fast, only 2-points guided.



Simulated annealing

Exploits an analogy with metallurgy/thermodynamics.

- a state of the system – a map m
- the energy of the state – the opposite of the loglike of the map

The probability of accepting a state of increasing energy is determined by the Boltzmann's distribution.

We start at an initial temperature T_i from an initial map (state) m with an energy (opposite of loglike) $-\ell(m)$ and slowly cool down the system while perturbating it til it reaches T_{min}.

Simulated annealing parameters

`annealing` *NbTries* T_i $T_{\min}$ α

1. T_i can be chosen arbitrarily (automatically adjusted).
2. *LPlateau* should be larger than $\frac{n.n-1}{2}$
3. α is close to 1. This fixes the length of the search (fast/slow cooling)
4. $T_{\min}$ should be small enough to avoid a premature end.

Play with the parameters α and $T_{\min}$. No definitive methodology to set them up.

Taboo search (greedy)

greedy *NbLoop NbExtra TabooMin TabooMax*

Starting from the best map in the heap and for a given number of steps:

1. Move to the best 2-OPT neighbor
2. unless this move is **taboo** (has been used recently, *TabooMin*=1 *TabooMax*=20, percentages)
3. one can nevertheless violate the taboo if the move improves over the best known solution.

This can be repeated several (*NbLoop*) times.

Final points

Additional facilities

- Able to tackle very large data sets (thousand of markers)
- Graphical interface with map display/print (all commands accessible through the shell interface in the GUI).
- Full user documentation, Open source code with optional LKH code.
- Includes a complete interpreted programming langage (for developping mapping strategies and reusing them).
- Available under Linux, Solaris and Windows.

The software web site:
http://www.inra.fr/mia/T/Carthagene