Supply Chain Reconstruction

Aaron Schankler

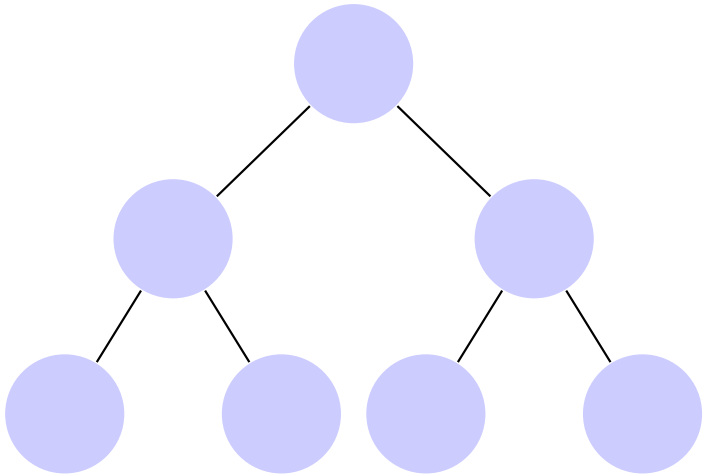
July 17, 2015

The Problem

- The co-occurrence of cutting agents shows "a promising potential in an intelligence perspective for analysing the local distribution process of heroin" (Terrettaz-Zufferey et al.)
- We want to recover information about the structure of the supply chain, such as which seizures are more closely related.
- We want to use only information about the end product.
- To this end, we propose a new model.

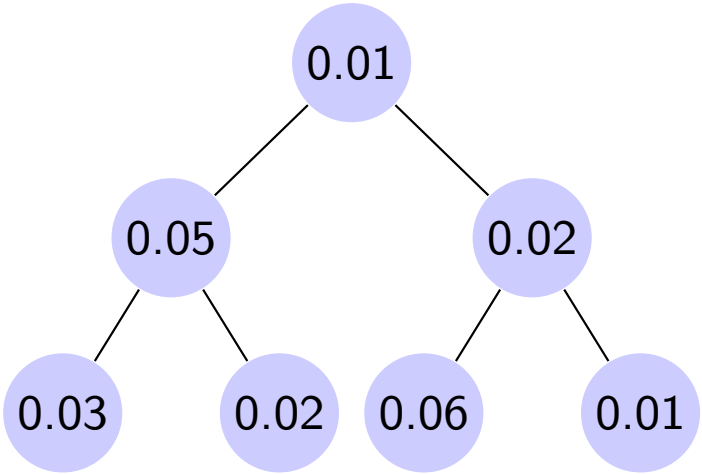
Our Model

Our Model



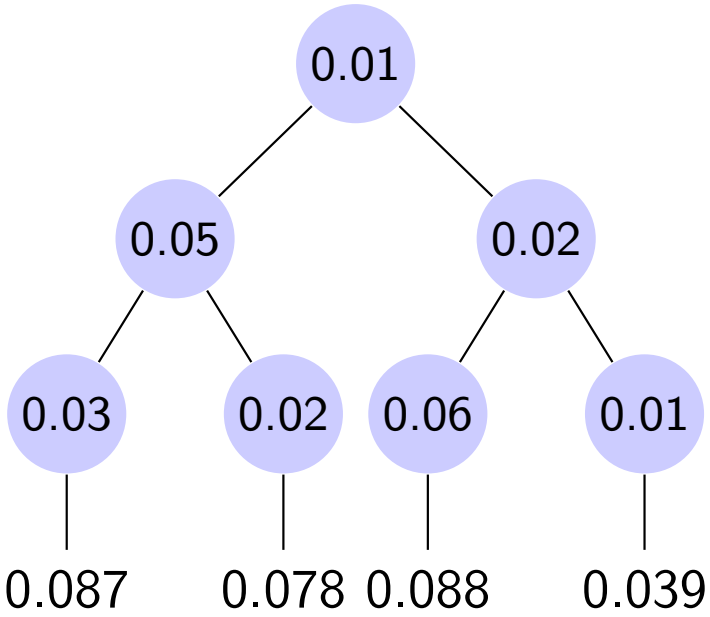
- We represent the supply chain as a tree where each node represents a supplier.

Our Model



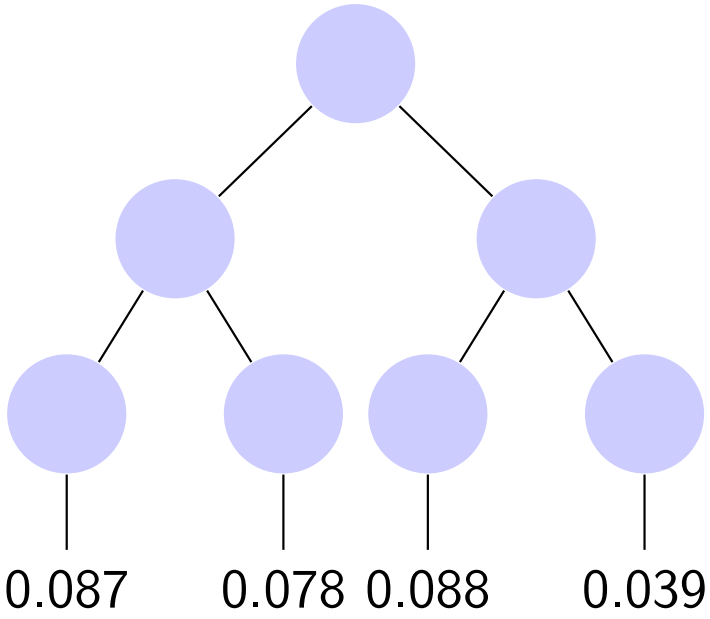
- We represent the supply chain as a tree where each node represents a supplier.
- Each supplier adds cutting agents in a fixed proportion.

Our Model



- We represent the supply chain as a tree where each node represents a supplier.
- Each supplier adds cutting agents in a fixed proportion.
- We know the total proportion of cutting agents at the lowest distribution level.

Our Model



- We represent the supply chain as a tree where each node represents a supplier.
- Each supplier adds cutting agents in a fixed proportion.
- We know the total proportion of cutting agents at the lowest distribution level.
- We do *not* know the proportion of cutting agents added at any node.

Our Model

- We represent the supply chain as a tree where each node represents a supplier.
- Each supplier adds cutting agents in a fixed proportion.
- We know the total proportion of cutting agents at the lowest distribution level.
- We do *not* know the proportion of cutting agents added at any node.
- We do *not* know how each sample is related to the others.

Assumptions

- Each supplier uses at least one cutting agent.
- A supplier will always add cutting agents in the same way (no time dependency).
- We know the proportion of cutting agents present at *all* of the leaves.
- Our goal is to recover the position of the known nodes in the original tree.
  - This information will allow us to reconstruct sibling relationships.

Our Method

- We first guess a tree shape.
- Based on the shape of the tree:
  - We can create a set of expressions that correspond to the proportion of each cutting agent present at each leaf;
  - These expressions depend on parameters that correspond to the proportion of each agent added at each node.
- We guess an arrangement of leaves and based on this guess pair each leaf value with the corresponding expression.
- We attempt to solve the resulting system; a solvable system should correspond to a possible configuration of the supply chain.

Solving the Systems

- The systems are under-constrained and non-linear, and so cannot be solved explicitly.
- We search numerically for parameters that come close to a solution.
- Our method is a specialized version of a technique called simulated annealing.

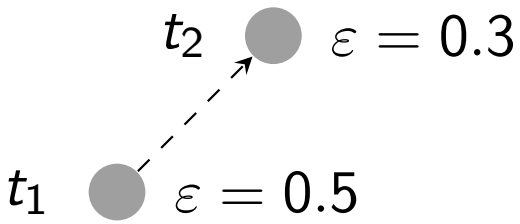
Solving the Systems

- The systems are under-constrained and non-linear, and so cannot be solved explicitly.
- We search numerically for parameters that come close to a solution.
- Our method is a specialized version of a technique called simulated annealing.

- Simulation begins at a random guess

Solving the Systems

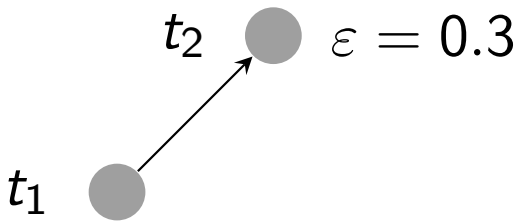
- The systems are under-constrained and non-linear, and so cannot be solved explicitly.
- We search numerically for parameters that come close to a solution.
- Our method is a specialized version of a technique called simulated annealing.



- Simulation begins at a random guess
- Take a random step from this guess and compare the errors (ε) at both points

Solving the Systems

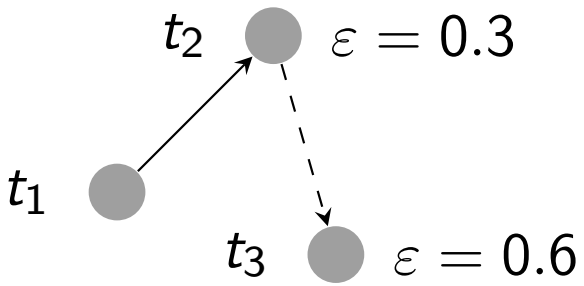
- The systems are under-constrained and non-linear, and so cannot be solved explicitly.
- We search numerically for parameters that come close to a solution.
- Our method is a specialized version of a technique called simulated annealing.



- Simulation begins at a random guess
- Take a random step from this guess and compare the errors (ε) at both points
- If the error of the new point is smaller, it is accepted as the new guess

Solving the Systems

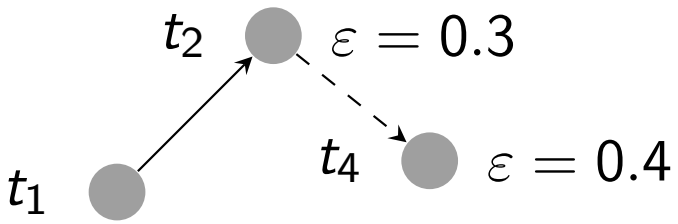
- The systems are under-constrained and non-linear, and so cannot be solved explicitly.
- We search numerically for parameters that come close to a solution.
- Our method is a specialized version of a technique called simulated annealing.



- Simulation begins at a random guess
- Take a random step from this guess and compare the errors (ε) at both points
- If the error of the new point is smaller, it is accepted as the new guess
- If error is larger, we randomly choose to accept or reject it

Solving the Systems

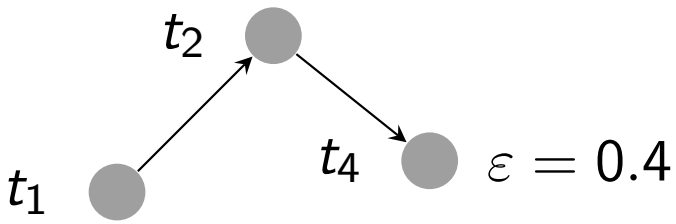
- The systems are under-constrained and non-linear, and so cannot be solved explicitly.
- We search numerically for parameters that come close to a solution.
- Our method is a specialized version of a technique called simulated annealing.



- Simulation begins at a random guess
- Take a random step from this guess and compare the errors (ε) at both points
- If the error of the new point is smaller, it is accepted as the new guess
- If error is larger, we randomly choose to accept or reject it

Solving the Systems

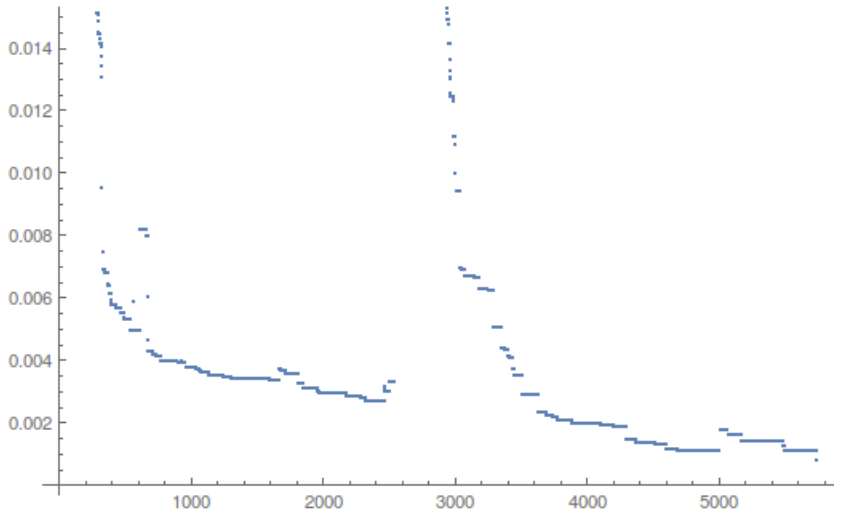
- The systems are under-constrained and non-linear, and so cannot be solved explicitly.
- We search numerically for parameters that come close to a solution.
- Our method is a specialized version of a technique called simulated annealing.



- Simulation begins at a random guess
- Take a random step from this guess and compare the errors (ε) at both points
- If the error of the new point is smaller, it is accepted as the new guess
- If error is larger, we randomly choose to accept or reject it

Solving the Systems

- The systems are under-constrained and non-linear, and so cannot be solved explicitly.
- We search numerically for parameters that come close to a solution.
- Our method is a specialized version of a technique called simulated annealing.



Implementation and Testing

- We have not yet made use of data on drug supply chains
  - Instead, we pick random parameters for each node.
  - We can then calculate plausible values for leaves to test the program on.
- Our method only tells us when a solution is possible, not when it is impossible.
  - We reject if a configuration is taking 'too long' to solve.
  - It is more effective to do multiple trials with a lower rejection threshold.
  - Testing bad configurations is more expensive than testing good ones.
- When testing our method, we look for:
  - Whether it can solve a correct tree
  - Whether it rejects incorrect trees

Results

- The effectiveness is affected by the shape of the tree and the number of cutting agents considered.
- Increasing the depth of the tree makes the system less constrained.
- Increasing the number of children at each node does not make the system less constrained.
- Increasing the number of cutting agents makes the system more constrained.
  - There are many more parameters, so systems are harder to solve.
  - Considering more agents *significantly* increases differentiation.
- We can eliminate many incorrect configurations on small trees.

Conclusion

- We have developed a new model that takes into account more quantitative information about the supply chain.
- We show that for small cases, our model can be used to gain information about the supply chain.
- Work left to do:
  - Do tests on guessing the tree shape, not just the configuration of leaves
  - Improve the SA implementation to reliably solve more complex trees
  - Find other methods to reject configurations without resorting to SA
  - Look at empirical data to see if our model is plausible