Metaheuristics for Search Problems in Genomics
— New Algorithms and Applications —

Stefano Benedettini[1]

DEIS, *Alma Mater Studiorum* Università di Bologna, Campus of Cesena, Italy
s.benedettini@unibo.it

Outline

1. Introduction
2. The Haplotype Inference Problem
3. The Founder Sequence Reconstruction Problem
4. Boolean Network Design

Outline

1. Introduction
2. The Haplotype Inference Problem
3. The Founder Sequence Reconstruction Problem
4. Boolean Network Design

Outline

1. Introduction
2. The Haplotype Inference Problem
3. The Founder Sequence Reconstruction Problem
4. Boolean Network Design

Outline

1. Introduction
2. The Haplotype Inference Problem
3. The Founder Sequence Reconstruction Problem
4. Boolean Network Design

Outline

1. Introduction
2. The Haplotype Inference Problem
3. The Founder Sequence Reconstruction Problem
4. Boolean Network Design

Motivations

- Outstanding goals in modern engineering:
  - Develop methodologies and tools to:
    - synthesise models of biological systems
    - analyse real or artificial biological systems
  - Oftentimes, such systems are complex
    - *Divide et impera* approach fails to capture important relationships
- Our objective:
  - Apply automatic procedures to the problem of model design
    - More precisely, we are interested in *model instantiation*
  - These procedures belong to the class of *search methods*

Definitions

Model: the set of entities and relationships that:

1. Can be used to explain a class of phenomena or systems
2. Can be expressed in a formal language

Model instance: application of aforementioned entities and concepts to describe a specific system

Predator-Prey model

Model: a system of (parametric) differential equations, such as the Lotka-Volterra equations

Model instance: their applications to describe population dynamics of rabbits and foxes in a forest

- How to instantiate parameters?

Our Proposed Methodology

- Concepts involved:
  - Employ metaheuristic techniques to instantiate models in genomics
  - A solution is a model instance
  - The search process manipulates (one or more) solutions
  - A merit factor (objective function) measures the “quality” of a models/solution with respect to a set of desiderata
- Applications:
  - Resolution of biological problems
  - Automatic synthesis of biological model instances (*Ensemble Approach*)

Metaheuristics for Solving Biological Problems

Facts

- Solutions to problems in genomics have to be “realistic”
  - They have to make sense for biologists
- Whatever model of a phenomenon is never 100% accurate by definition
- A complete technique spends time to return a proof of optimality

Motivations

- An optimal solution for an approximate model might not be useful
- Metaheuristics can easily incorporate different objective function components

Metaheuristics for Solving Biological Problems

Facts

- Solutions to problems in genomics have to be “realistic”
  - They have to make sense for biologists
- Whatever model of a phenomenon is never 100% accurate by definition
- A complete technique spends time to return a proof of optimality

Motivations

- An optimal solution for an approximate model might not be useful
- Metaheuristics can easily incorporate different objective function components

A Statistical Approach to Modeling

Ensemble Approach (EA)

- Let's define a feature space $\mathcal{F}$
- A generic system σ is identified by a coordinate vector $\Phi(\sigma) = \langle f_1, f_2, \ldots, f_n \rangle \in \mathcal{F}$
- Is a model M an accurate description of a system σ?
- The EA says that it is the case if point $\Phi(\sigma)$ is in the same cluster as (different realizations of) model M

Role of Metaheuristics

- How to find model instances close to σ in $\mathcal{F}$?
- Define a suitable optimization problem and solve it

A Statistical Approach to Modeling

Ensemble Approach (EA)

- Let's define a feature space $\mathcal{F}$
- A generic system σ is identified by a coordinate vector $\Phi(\sigma) = \langle f_1, f_2, \ldots, f_n \rangle \in \mathcal{F}$
- Is a model M an accurate description of a system σ?
- The EA says that it is the case if point $\Phi(\sigma)$ is in the same cluster as (different realizations of) model M

Role of Metaheuristics

- How to find model instances close to σ in $\mathcal{F}$?
- Define a suitable optimization problem and solve it

A Statistical Approach to Modeling

Ensemble Approach (EA)

- Let's define a feature space $\mathcal{F}$
- A generic system σ is identified by a coordinate vector $\Phi(\sigma) = \langle f_1, f_2, \ldots, f_n \rangle \in \mathcal{F}$
- Is a model M an accurate description of a system σ?
- The EA says that it is the case if point $\Phi(\sigma)$ is in the same cluster as (different realizations of) model M

Role of Metaheuristics

- How to find model instances close to σ in $\mathcal{F}$?
- Define a suitable optimization problem and solve it

Outline

1 Introduction

2 The Haplotype Inference Problem

3 The Founder Sequence Reconstruction Problem

4 Boolean Network Design

Biological Definition

Entities involved

- Diploid organism
- Haplotype
- Genotype

Issues in DNA Sequencing

- Haplotype collection is *very expensive*
- On the contrary, genotype collection *is not*

Haplotype Inference

- Obtain haplotype information explaining genotype data

Biological Definition

Entities involved

- Diploid organism
- Haplotype
- Genotype

Issues in DNA Sequencing

- Haplotype collection is *very expensive*
- On the contrary, genotype collection *is not*

Haplotype Inference

- Obtain haplotype information explaining genotype data

Biological Definition

Entities involved

- Diploid organism
- Haplotype
- Genotype

Issues in DNA Sequencing

- Haplotype collection is *very expensive*
- On the contrary, genotype collection *is not*

Haplotype Inference

- Obtain haplotype information explaining genotype data

Mathematical Definition

Definition (Genotype)

A genotype g is a vector in $\{0, 1, 2\}^m$

Definition (Haplotype)

An haplotype h is a vector $\{0, 1\}^m$

Definition (Genotype Resolution)

We say that the haplotypes h, l resolve genotype g, and we write $\langle h, l \rangle \triangleright g$, if and only if:

$$\begin{array}{lcl} g[i] = 0 & \Rightarrow & h[i] = l[i] = 0 \\ g[i] = 1 & \Rightarrow & h[i] = l[i] = 1 \\ g[i] = 2 & \Rightarrow & h[i] \neq l[i] \end{array}$$

Definition (Haplotype Inference by Maximum Parsimony)

Let G be a set of genotype; find the minimal set of haplotypes H so that $\forall g \in G, \exists\, h, l \in H \mid \langle h, l \rangle \triangleright g$

Mathematical Definition

Definition (Genotype)

A genotype g is a vector in $\{0, 1, 2\}^m$

Definition (Haplotype)

An haplotype h is a vector $\{0, 1\}^m$

Definition (Genotype Resolution)

We say that the haplotypes h, l resolve genotype g, and we write $\langle h, l \rangle \triangleright g$, if and only if:

$$\begin{array}{lcr} g[i] = 0 & \Rightarrow & h[i] = l[i] = 0 \\ g[i] = 1 & \Rightarrow & h[i] = l[i] = 1 \\ g[i] = 2 & \Rightarrow & h[i] \neq l[i] \end{array}$$

Definition (Haplotype Inference by Maximum Parsimony)

Let G be a set of genotype; find the minimal set of haplotypes H so that $\forall g \in G, \exists\, h, l \in H \mid \langle h, l \rangle \triangleright g$

Mathematical Definition

Definition (Genotype)

A genotype g is a vector in $\{0, 1, 2\}^m$

Definition (Haplotype)

An haplotype h is a vector $\{0, 1\}^m$

Definition (Genotype Resolution)

We say that the haplotypes h, l resolve genotype g, and we write $\langle h, l \rangle \triangleright g$, if and only if:

$$\begin{array}{lcr} g[i] = 0 & \Rightarrow & h[i] = l[i] = 0 \\ g[i] = 1 & \Rightarrow & h[i] = l[i] = 1 \\ g[i] = 2 & \Rightarrow & h[i] \neq l[i] \end{array}$$

Definition (Haplotype Inference by Maximum Parsimony)

Let G be a set of genotype; find the minimal set of haplotypes H so that
$\forall g \in G, \exists\, h, l \in H \mid \langle h, l \rangle \triangleright g$

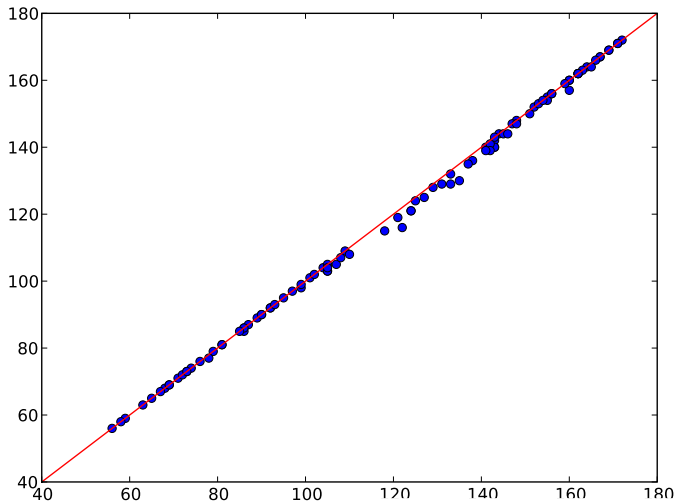
State of the Art

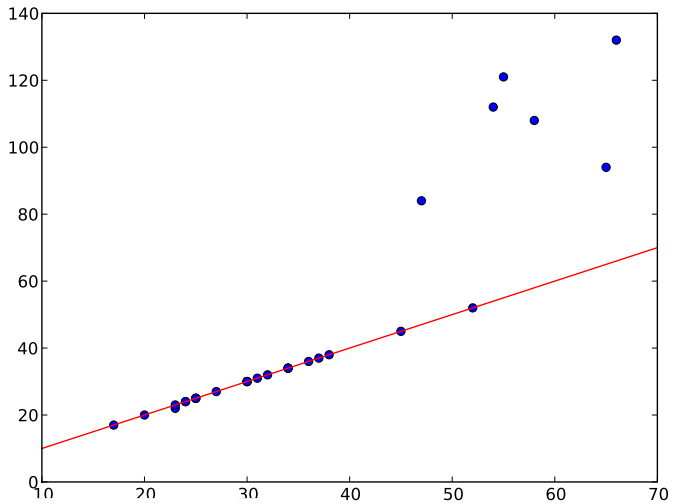
- Integer Linear Programming formulation
  - Branch-and-Bound, Branch-and-Cut, . . .
- Incomplete techniques
  - Genetic Algorithms
  - Tabu Search
- Pseudo-Boolean Optimization
  - `rpoly`
- Complete techniques don't scale well for large-sized instances

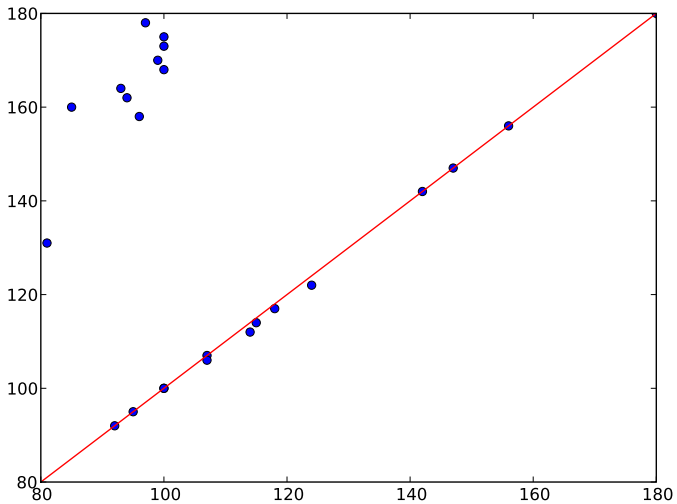
Contributions

- Hybrid Ant Colony Optimization algorithm
- Flexible:
  - Can accommodate different genetic models by changing the objective function
  - Can integrate different resolution criteria, e.g., statistical techniques
- Effective:
  - Comparable performance to state of the art exact solver `rpoly`
- Scalable:
  - Can cope with large instance size
  - Superior to `rpoly` in this cases

Experimental Results







Outline

1 Introduction

2 The Haplotype Inference Problem

3 The Founder Sequence Reconstruction Problem

4 Boolean Network Design

Biological Definition

Purpose

- Study the evolutionary history of a population of human DNA sequences
- Helps biologist to discover genetic bases of complex diseases

Genetic Basis

- A population is evolved from a relatively small number of haplotype founders
- Crossover breaks and shuffles fragments of sequences

Goal

Find a set of founders which can reconstruct the sequences in the population with the least number of crossovers

An Example of a “Mosaic”

0	1	1	0	1	0	0
1	1	0	1	1	1	1
1	0	1	0	0	0	1

1	1	0	1	1	0	1
1	0	1	0	0	0	1
0	1	1	1	1	1	1
0	1	1	0	1	0	0
1	1	0	0	0	1	1

- Breakpoints correspond to crossover events
- Solution value: 4
- Mosaic is not unique
- Given a founder matrix, the computation of the optimal mosaic is polynomial

State of the Art

- Complete algorithms:
  - Dynamic Programming
  - Tree search enhanced with pseudo Branch-and-Bound pruning: RecBlock
- Incomplete techniques:
  - Greedy Heuristic
  - Tabu Search
- Complete techniques don't scale well for large-sized instances
- Nevertheless, running times for Tabu Search are quite long and still performances are not impressive

State of the Art

- Complete algorithms:
  - Dynamic Programming
  - Tree search enhanced with pseudo Branch-and-Bound pruning: RecBlock
- Incomplete techniques:
  - Greedy Heuristic
  - Tabu Search
- Complete techniques don't scale well for large-sized instances
- Nevertheless, running times for Tabu Search are quite long and still performances are not impressive

State of the Art

- Complete algorithms:
  - Dynamic Programming
  - Tree search enhanced with pseudo Branch-and-Bound pruning: RecBlock
- Incomplete techniques:
  - Greedy Heuristic
  - Tabu Search
- Complete techniques don't scale well for large-sized instances
- Nevertheless, running times for Tabu Search are quite long and still performances are not impressive

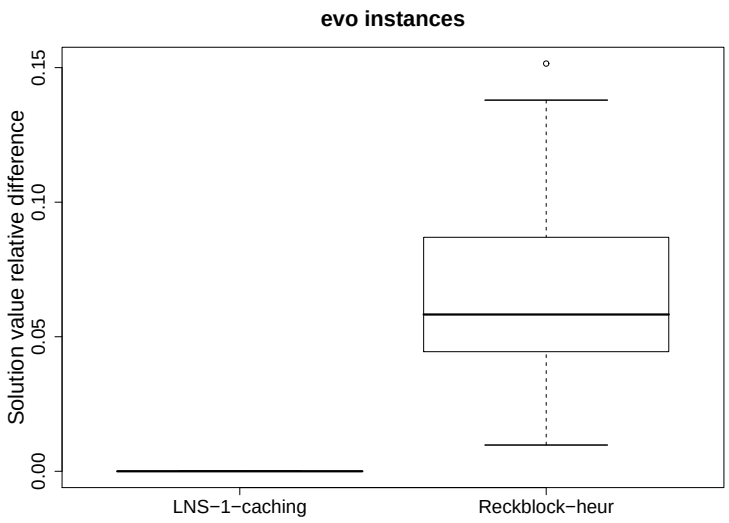
State of the Art

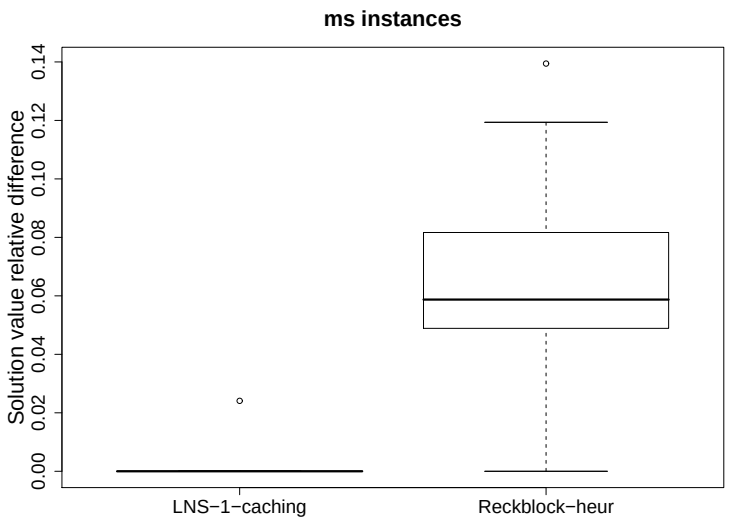
- Complete algorithms:
  - Dynamic Programming
  - Tree search enhanced with pseudo Branch-and-Bound pruning: RecBlock
- Incomplete techniques:
  - Greedy Heuristic
  - Tabu Search
- Complete techniques don't scale well for large-sized instances
- Nevertheless, running times for Tabu Search are quite long and still performances are not impressive

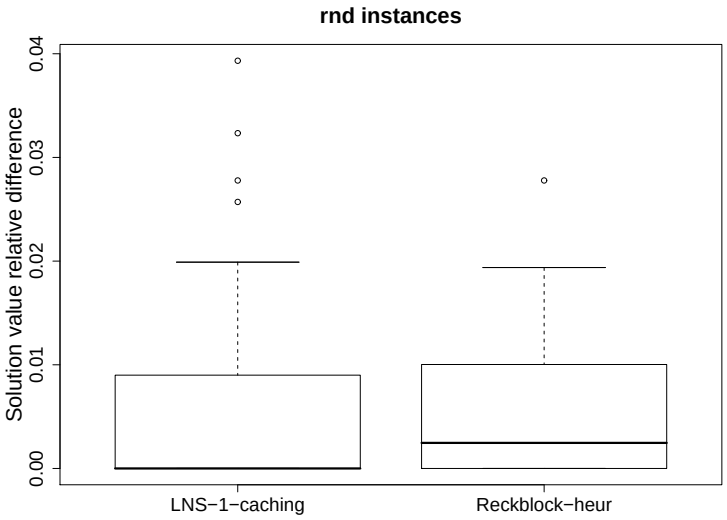
Contributions

- Randomized Iterated Greedy
  - Fast incomplete algorithm
  - Better than former Tabu Search algorithm
  - Provides the initial solution to our LNS
- Large Neighborhood Search (LNS)
  - Integrates and boosts RecBlock
  - Anytime solver
  - But eventually reaches the optimum if given enough time
  - Current state of the art method for FSRP

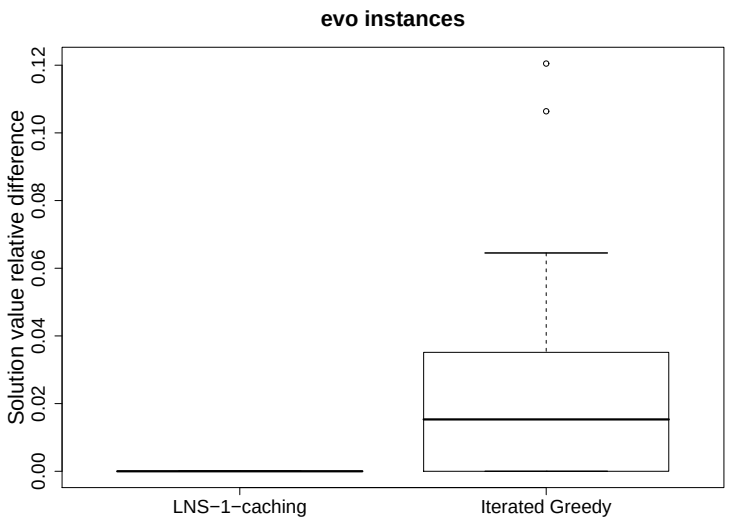
Comparison against RecBlock

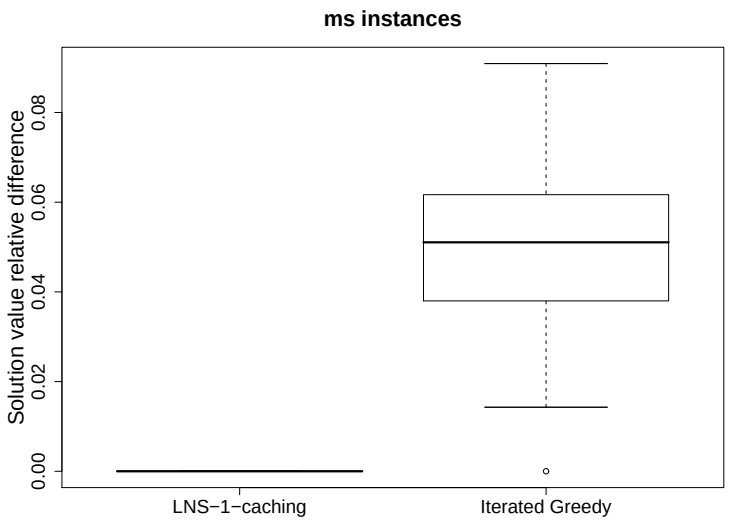


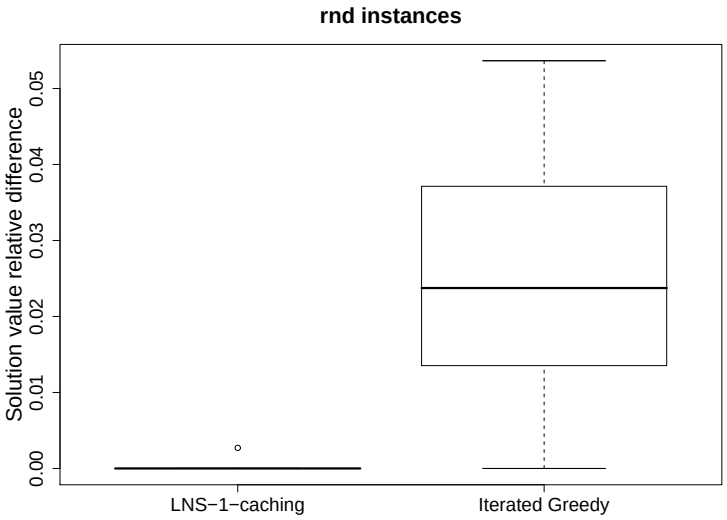




Comparison against Iterated Greedy







Outline

1 Introduction

2 The Haplotype Inference Problem

3 The Founder Sequence Reconstruction Problem

4 Boolean Network Design

Scope of the Research

- Boolean networks (BNs) are complex dynamical systems (and models of complex systems)
- Recent research, mainly in biology, needs to find/design models satisfying given requirements
- Hot topic in complex system biology

Biological Standpoint

- Employ Boolean networks as a modeling tool
- Ensemble Approach

Boolean Networks

- Introduced by Stuart Kauffman as a models of genetic regulatory networks (GRNs)
- Discrete-time/discrete-state dynamical system
- Non trivial (*complex*) dynamics

Boolean Networks

Structure

- Directed graph of N nodes
- Node i:
  - Boolean value x_i
  - Boolean function f_i
- Boolean function arguments are variables associated to input nodes of i
- Node state (i.e., Boolean variable) updated as a function of f_i

Boolean Networks

Dynamics

- System state at time t: $s(t) = (x_1(t), \ldots, x_N(t))$
- Dynamics controls node update
- Deterministic synchronous update
- Every state has an unique successor

Variants

- Several variants exist:
  - BNs with asynchronous dynamics
  - Boolean threshold networks
  - Probabilistic Boolean networks
  - Glass networks
- We focus on the most studied model

Dynamical Features

Trajectories

- Transient
- *Attractor*

Attractors

- Fixpoints
- Cycles

Basin of Attraction (of attractor $\mathcal{A}$)

Set of states belonging to the trajectories ending at attractor $\mathcal{A}$

Problem Difficulties

- How do we obtain networks that match a set of desiderata?
  - Network space is enormous so random generation is not an option
- How do we effectively explore such space?
- How do we guide such search process?
- How do we evaluate the “quality” of a network?

Our Approach to Automatic Design

- We cast this problem into the framework of combinatorial optimization
  1. A Boolean network is a point in the search space
  2. A suitable objective function (OF) is defined
  3. Search space is equipped with a notion of neighborhood (topology)
  4. We choose and apply a (meta)heuristic search strategy
  5. OF evaluation can be performed by sampling the network state space

Iterated Local Search Framework

Generic Algorithm Template

INPUT: a local search
$s \leftarrow$ generateInitialSolution()
$s^* \leftarrow$ localSearch(s) {Stochastic Descent in our experiments}
while termination conditions not met **do**
  $s' \leftarrow$ perturbation(s_{best})
  $s'_{ls} \leftarrow$ localSearch(s')
  $s^* \leftarrow$ acceptanceCriterion(s^*, s'_{ls})
end while
return s^*

The Boolean Network Toolkit

- OF evaluation is the most computationally intensive task
- A fast and flexible simulator is required

BnToolkit

- Efficient library BN simulation and analysis
- Written in C++
- Open Source project
- Available at `http://booleannetwork.sourceforge.net`

The Boolean Network Toolkit

- OF evaluation is the most computationally intensive task
- A fast and flexible simulator is required

BnToolkit

- Efficient library BN simulation and analysis
- Written in C++
- Open Source project
- Available at `http://booleannetwork.sourceforge.net`

Applications

- Our design methodology successfully applied to three problems in biological modeling
- BNs with maximally distant attractors
  - We can study properties of more biologically plausible networks
- Boolean networks as classifiers
  - Investigate an important topic in artificial learning systems
- BNs as models of cellular differentiation
  - It helps to validate the model against real data
  - We aim to generate networks that predict behaviour of existing cell types

BNs with Maximally Distant Attractors

Limitations of Classic BNs

- Attractors in BNs can be interpreted as cell types
- The attractor set in classic BNs is very similar attractors (they differ for just a few values)
- They are no longer distinguishable if a different update scheme is used
- Synchronous deterministic update could generate spurious attractors

Overcoming the Limitations

- We aim at designing synchronous deterministic BNs in which attractors be as much different as possible

BNs with Maximally Distant Attractors

Limitations of Classic BNs

- Attractors in BNs can be interpreted as cell types
- The attractor set in classic BNs is very similar attractors (they differ for just a few values)
- They are no longer distinguishable if a different update scheme is used
- Synchronous deterministic update could generate spurious attractors

Overcoming the Limitations

- We aim at designing synchronous deterministic BNs in which attractors be as much different as possible

Boolean Networks as Classifiers

Biological motivations

- Cell behaviour can change in response to different conditions in the environment
- From an abstract standpoint, a cell is able to solve a classification problem
  - Environmental conditions are examples to classify
  - Cell dynamics, represented by attractor states, are responses

Goal

- Design BNs which are able to solve the Density Classification Problem (DCP)
  - Determine if a binary string contains more 0s than 1s

Boolean Networks as Classifiers

Biological motivations

- Cell behaviour can change in response to different conditions in the environment
- From an abstract standpoint, a cell is able to solve a classification problem
  - Environmental conditions are examples to classify
  - Cell dynamics, represented by attractor states, are responses

Goal

- Design BNs which are able to solve the Density Classification Problem (DCP)
  - Determine if a binary string contains more 0s than 1s

A Model of Cellular Differentiation

Biological motivations

- Overcoming the limitation of attractors/cell types correspondence
- Cell types are Threshold Ergodic Sets:
  - Sets of attractors...
  - Stable under certain level of noise
  - Noise models external environmental conditions
- Can describes differentiation trees of pluripotent cells

Goal

- Impose constraints on:
  - Attractors landscapes
  - Differentiation tree shapes
- Ongoing research

A Model of Cellular Differentiation

Biological motivations

- Overcoming the limitation of attractors/cell types correspondence
- Cell types are Threshold Ergodic Sets:
  - Sets of attractors...
  - Stable under certain level of noise
  - Noise models external environmental conditions
- Can describes differentiation trees of pluripotent cells

Goal

- Impose constraints on:
  - Attractors landscapes
  - Differentiation tree shapes
- Ongoing research

A Model of Cellular Differentiation

Biological motivations

- Overcoming the limitation of attractors/cell types correspondence
- Cell types are Threshold Ergodic Sets:
  - Sets of attractors...
  - Stable under certain level of noise
  - Noise models external environmental conditions
- Can describes differentiation trees of pluripotent cells

Goal

- Impose constraints on:
  - Attractors landscapes
  - Differentiation tree shapes
- Ongoing research

Summary

- Competitive hybrid metaheuristic algorithm for Haplotype Inference by Parsimony
- State-of-the-art metaheuristic algorithm for the Founder Sequence Reconstruction Problem
- Automatic design methodology of BNs successfully applied to three modeling problems
- Flexible and efficient Boolean network simulator software

Activity I

- Journal papers:
  - Battarra, M., Benedettini, S., and Roli, A. (2011). Leveraging saving-based algorithms by master-slave genetic algorithms. *Engineering Applications of Artificial Intelligence*, 24:555–566
  - Roli, A., Benedettini, S., Stützle, T., and Blum, C. (2012). Large neighbourhood search algorithms for the founder sequence reconstruction problem. *Computers & Operations Research*, 39(2):213–224
  - Benedettini, S., Manfroni, M., Villani, M., Serra, R., Gagliardi, A., Pinciroli, C., Birattari, M., and Roli, A. (accepted with minor revision). Learning Boolean networks: an approach with metaheuristics. *Neurocomputing*

Activity II

- Collaborations:
  - Prof. Christian Blum, UPC, Barcelona, from sep. 2009 to mar. 2010
  - Prof. Thomas Stützle, IRIDIA, ULB, Brussels, from sep. 2010 to dec. 2010
  - Prof. Roberto Serra and Marco Villani, University of Modena and Reggio Emilia, 2009-ongoing

Metaheuristics for Search Problems in Genomics
— New Algorithms and Applications —

Stefano Benedettini[1]

DEIS, *Alma Mater Studiorum* Università di Bologna, Campus of Cesena, Italy
s.benedettini@unibo.it

References I

[1] Battarra, M., Benedettini, S., and Roli, A. (2011). Leveraging saving-based algorithms by master-slave genetic algorithms. *Engineering Applications of Artificial Intelligence*, 24:555–566.

[2] Benedettini, S., Manfroni, M., Villani, M., Serra, R., Gagliardi, A., Pinciroli, C., Birattari, M., and Roli, A. (accepted with minor revision). Learning Boolean networks: an approach with metaheuristics. *Neurocomputing*.

[3] Roli, A., Benedettini, S., Stützle, T., and Blum, C. (2012). Large neighbourhood search algorithms for the founder sequence reconstruction problem. *Computers & Operations Research*, 39(2):213–224.