

Towards automated selection of parts for genetic regulatory networks

(BBN) *F. Yaman*, A. Adler, J. Beal

(BU) S. Bhatia, D. Densmore

(MIT) R. Weiss, N. Davidsohn

Work sponsored by DARPA I2O under contract HR0011-10-C-0168; the views and conclusions contained in this document are those of the authors and not DARPA or the U.S. Government.

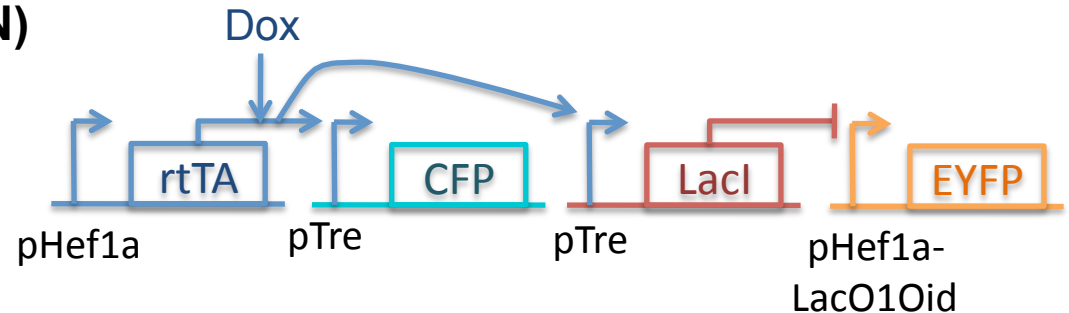


Raytheon
BBN Technologies

Designing Transcriptional Networks

Genetic Regulatory Network (GRN)

- Specific design
- Ready to assemble



Behavior description

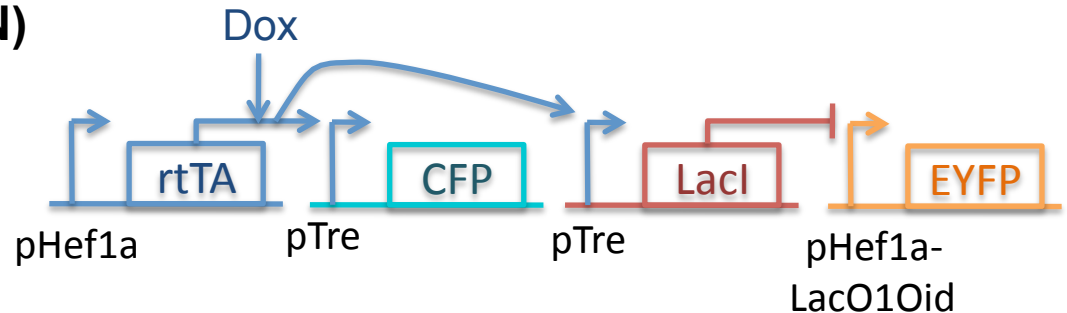
- High-level design
- Must be mapped to biology

```
if (sense(Dox))  
  then fluoresce(cyan)  
  else fluoresce(yellow)
```

Abstract Genetic Regulatory Networks

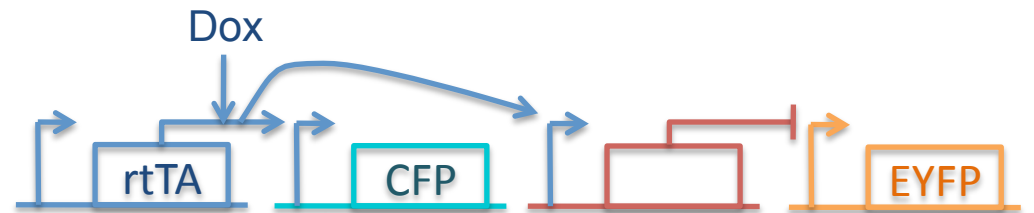
Genetic Regulatory Network (GRN)

- Specific design
- Ready to assemble



Abstract Genetic Regulatory Network (AGRN)

- Template representing multiple GRNs
- Contains necessary constraints



Behavior description

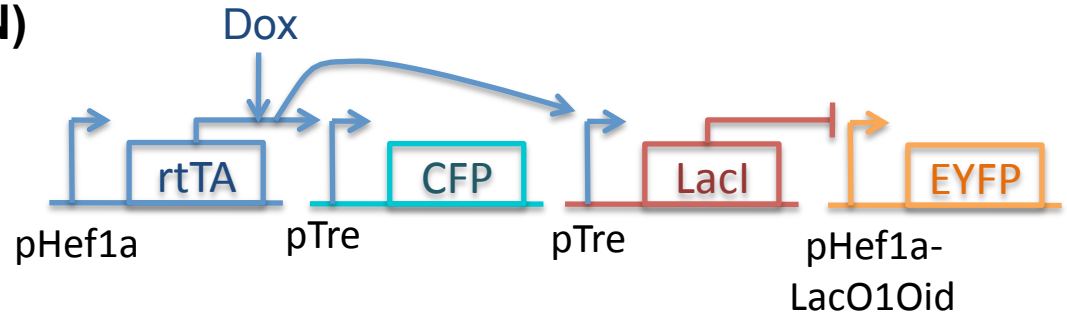
- High-level design
- Must be mapped to biology

```
if (sense(Dox))  
  then fluoresce(cyan)  
  else fluoresce(yellow)
```

From AGRNs to GRNs

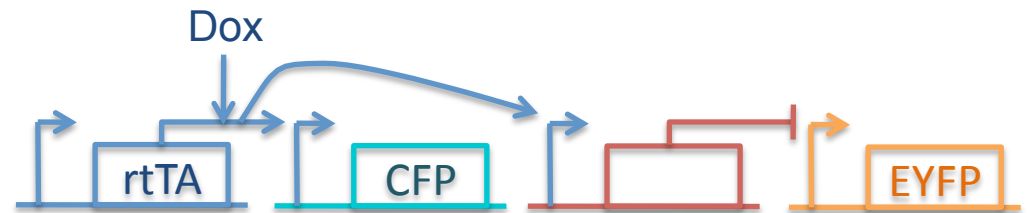
Genetic Regulatory Network (GRN)

- Specific design
- Ready to assemble



Abstract Genetic Regulatory Network (AGRN)

- Template representing multiple GRNs
- Contains necessary constraints



How can we map the abstract parts in an AGRN to real parts?

Solution: Feature Mapping + Signal Matching

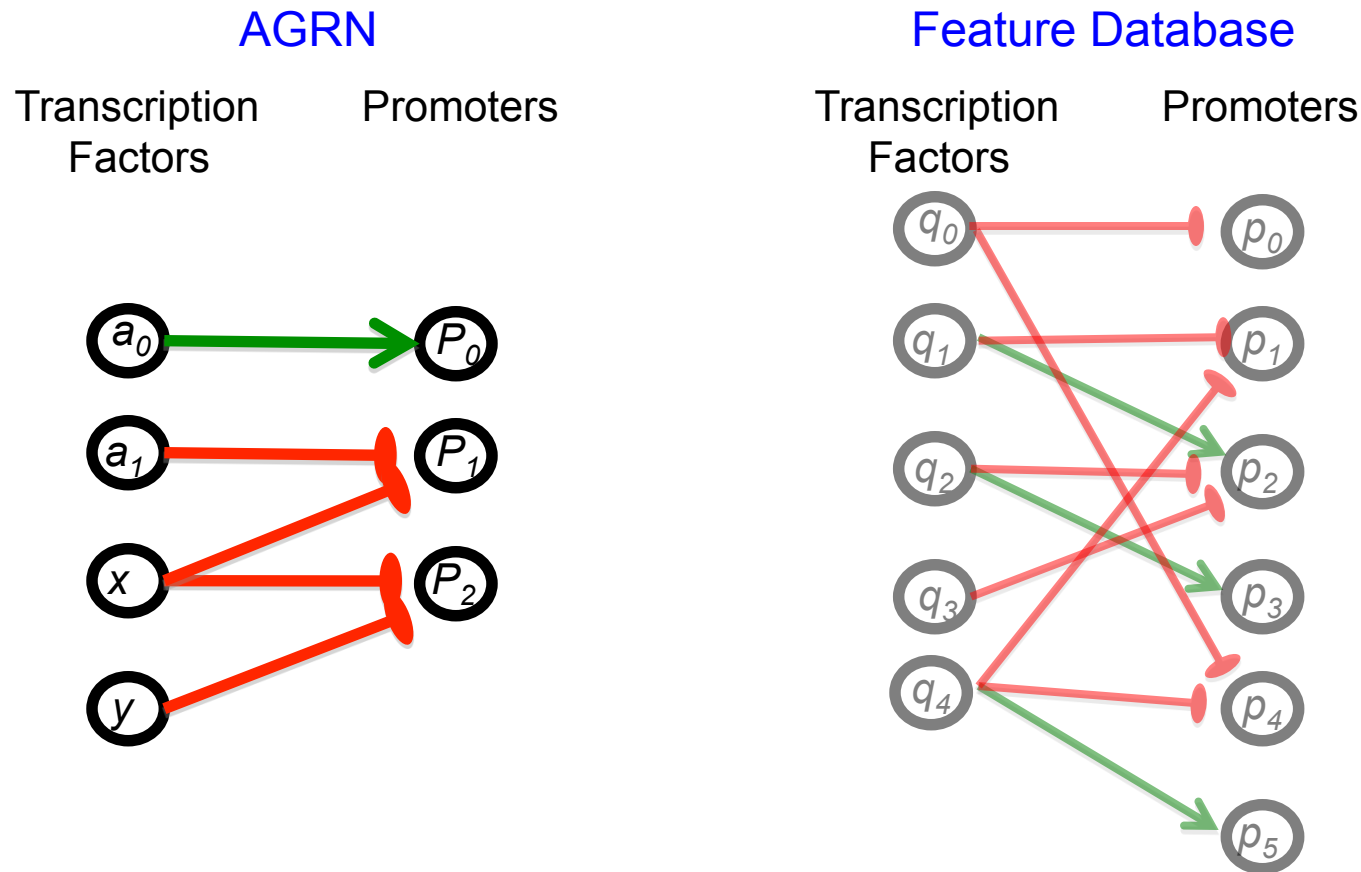
Feature Mapping

- **Feature**: a DNA sequence responsible for a specific biochemical behavior, e.g., promoter
- **Feature database**: a collection of features with the regulatory relationships between them



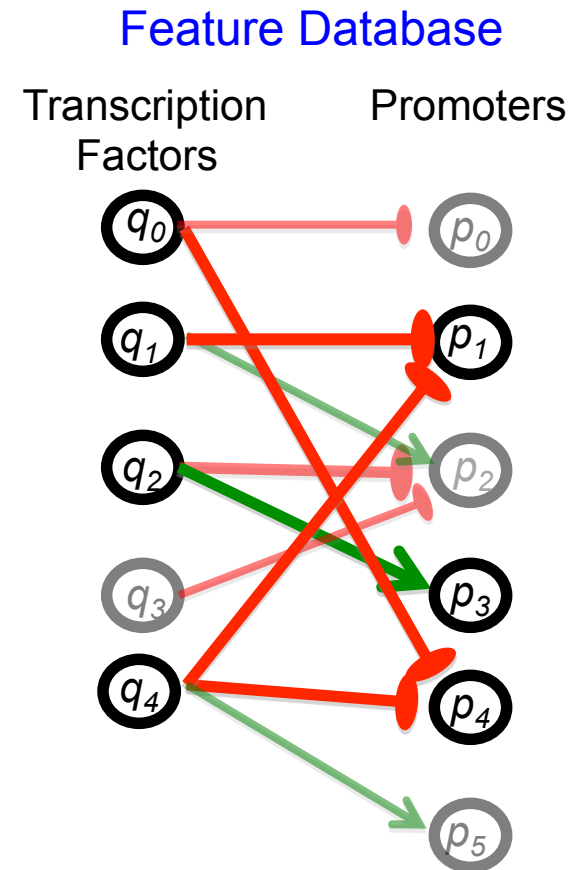
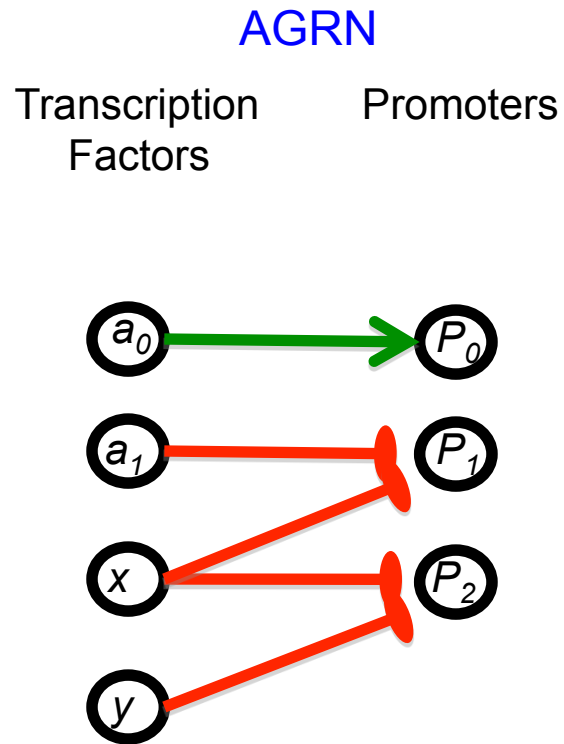
- **Feature mapping**: assign features from the database to the variables in AGRN
 - The mapping should satisfy all edges in AGRN and not entail additional interactions.
 - Given an AGRN G and a feature database H , find a network of promoters and transcription factors in H that is isomorphic to G .

Feature Mapping: assign features to variables



- **Feature mapping:** Given bipartite graphs G and H , find a subgraph of H that is strictly isomorphic to G .

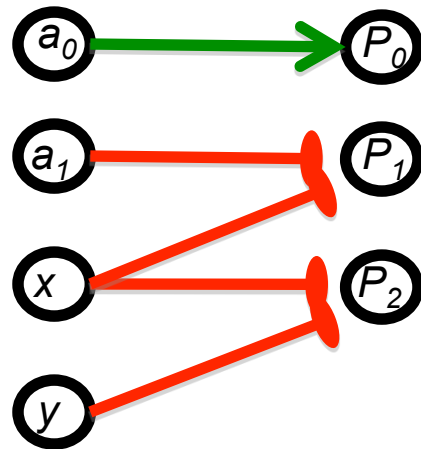
Feature Mapping: assign features to variables



Feature Mapping: assign features to variables

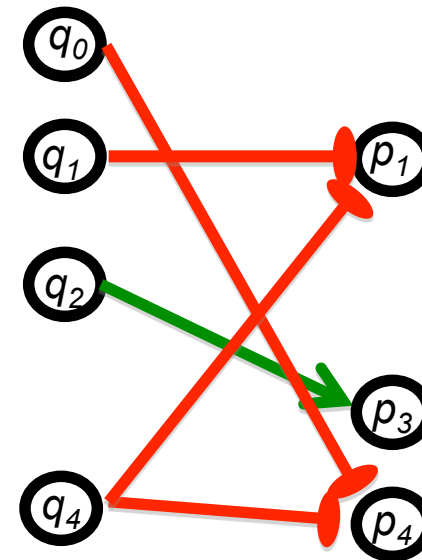
AGRN

Transcription Factors Promoters

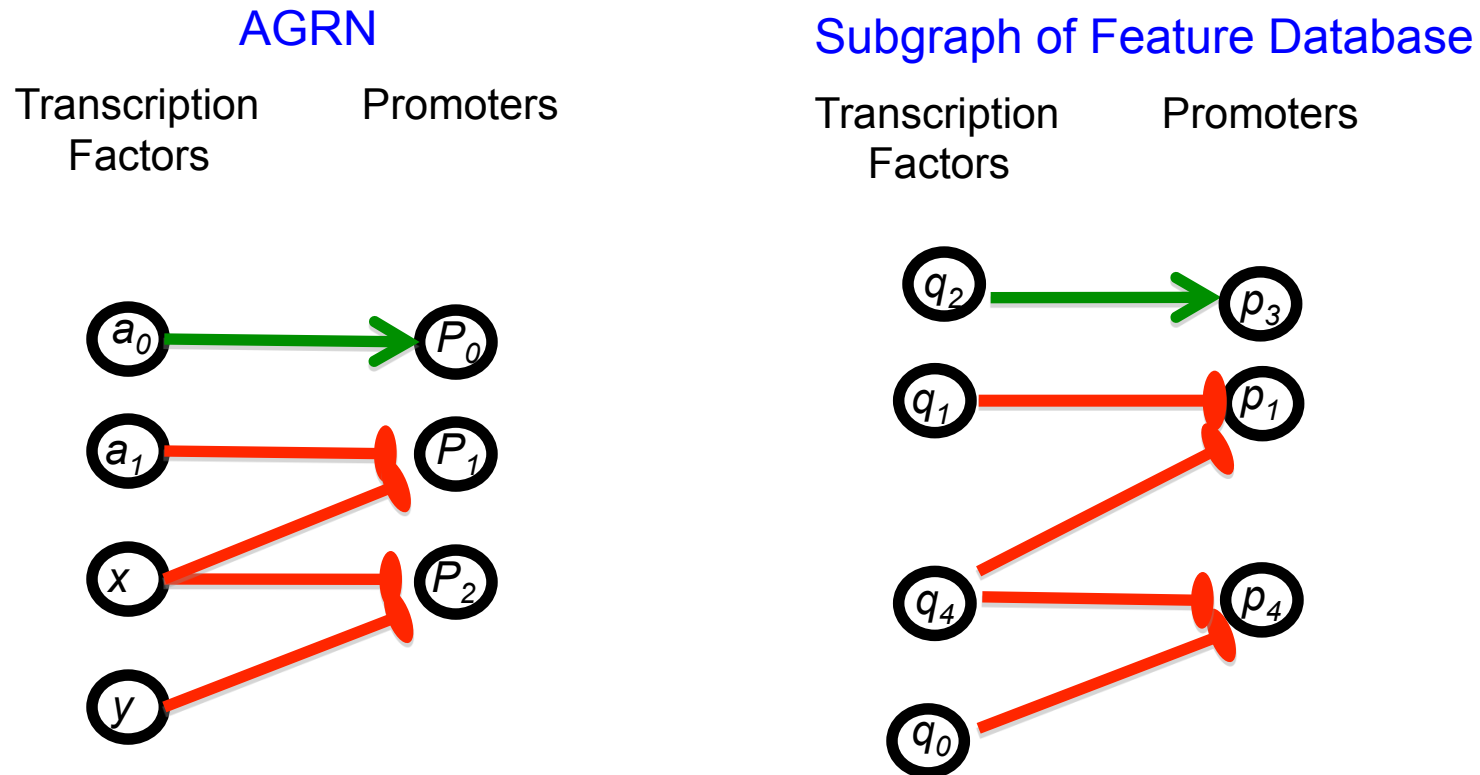


Subgraph of Feature Database

Transcription Factors Promoters



Feature Mapping: assign features to variables



- This problem is NP-complete: There is no fast algorithm for solving every instance of this problem*.

Solution: Heuristic guided search

```
Solve (AGRN, FDB, Assignments)
  Pick a node v in AGRN that is not in Assignments
  If no such node exists
    Return Assignments
  For every node f in FDB do
    NewAssignments = Assignments + (v,f)
    result =Solve (AGRN, FDB, NewAssignments )
    If result is not FAIL
      Return result
  Return FAIL;
```

Solution: Heuristic guided search

```
Solve (AGRN, FDB, Assignments)
  Pick a node v in AGRN that is not in Assignments
  If no such node exists
    Return Assignments
  For every node f in FDB do
    NewAssignments = Assignments + (v, f)
    result = Solve (AGRN, FDB, NewAssignments )
    If result not FAIL
      Return result
  Return
```

Same type as v
Have at least same or more in/out degree as v
Can support every edge of v

Solution: Heuristic guided search

Rank the nodes: small domain --> higher rank
Pick the highest ranking node v

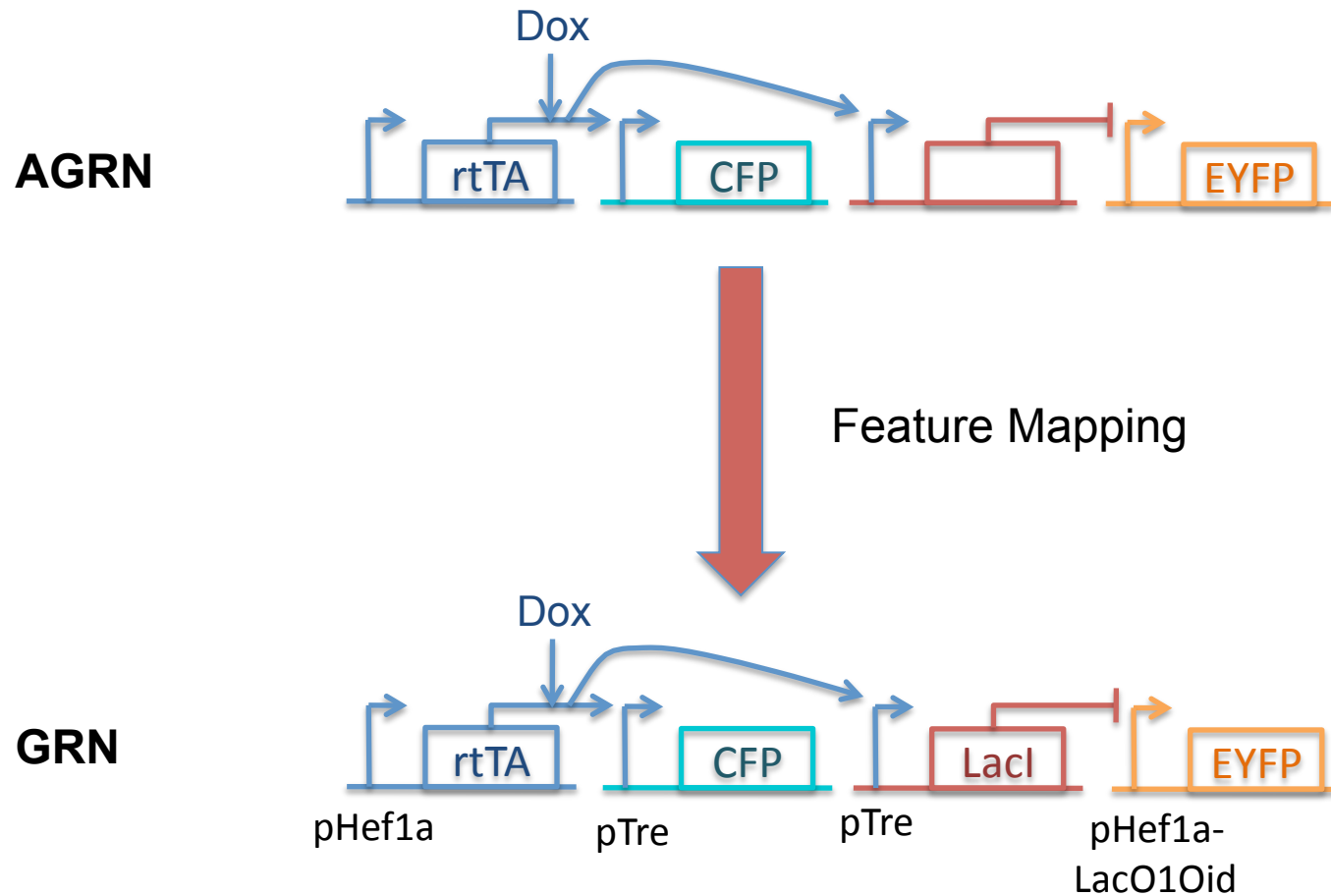
```
Solve (AGRN, FDB, Assignments)
  Pick a node  $v$  in AGRN that is not in Assignments
  If no such node exists
    Return Assignments
  For every node  $f$  in FDB do
    NewAssignments = Assignments + ( $v, f$ )
    result = Solve (AGRN, FDB, NewAssignments )
    If result is not FAIL
      Return result
  Return FAIL;
```

Solution: Heuristic guided search

```
Solve (AGRN, FDB, Assignments)
  Pick a node v in AGRN that is not in Assignments
  If no such node exists
    Return Assignments
  For every node f in FDB do
    NewAssignments = Assignments + (v, f)
    result = Solve (AGRN, FDB, NewAssignments )
    If result is not FAIL
      Return result
  Return FAIL;
```

Rank nodes f; (appears in more variable domains --> higher rank)
Try the nodes in ranking order

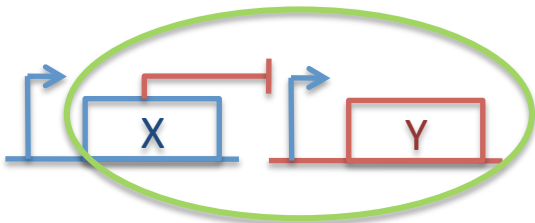
Feature Mapping produces a GRN



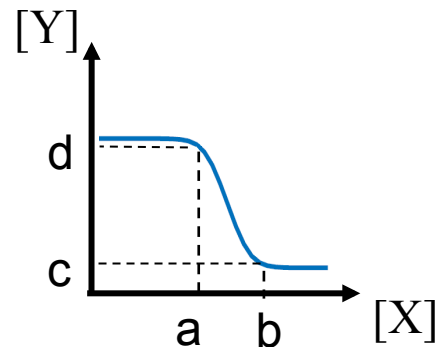
Signal Matching

- Feature mapping ensures the pair wise logical relationships but there is also signal ranges to consider

BioDevice



Transfer Curve

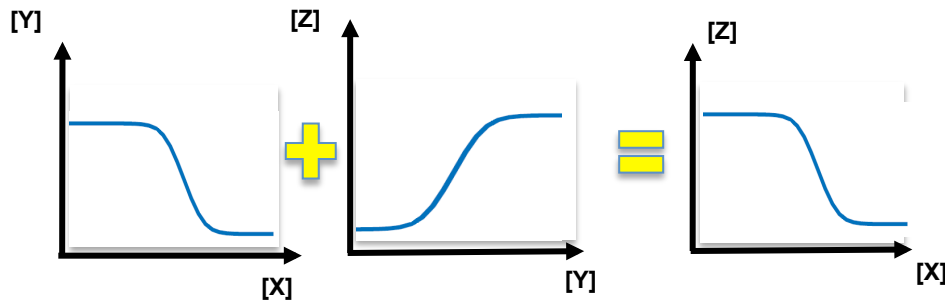
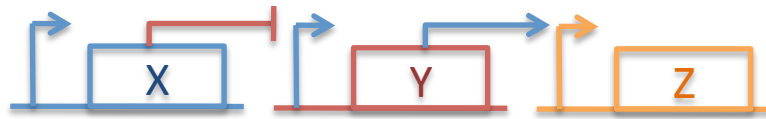


Digital Behavior

X	Y
Off [x] < a	On [y] > d
On [x] > b	Off [y] < c

Signal Matching

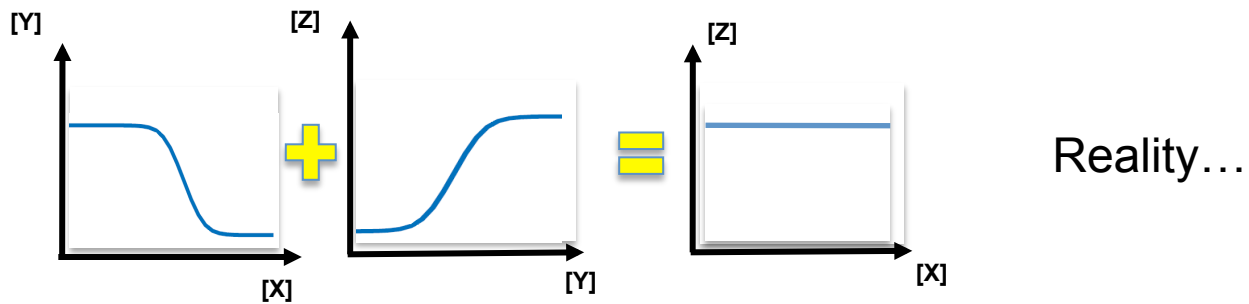
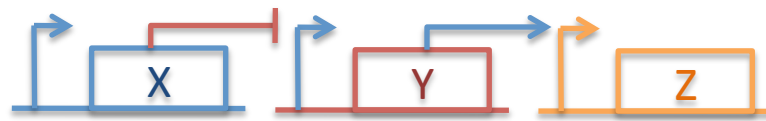
- Composition should preserve digital behavior



Ideally...

Signal Matching

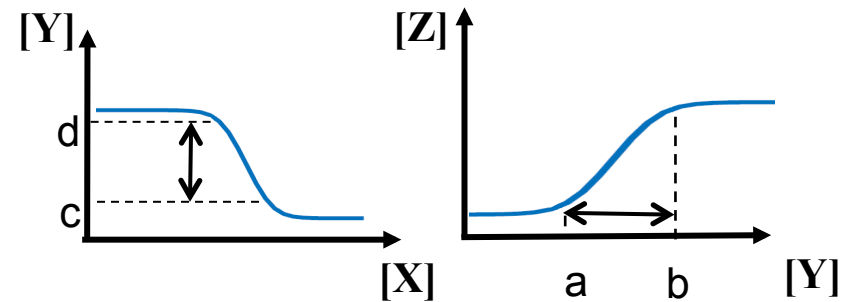
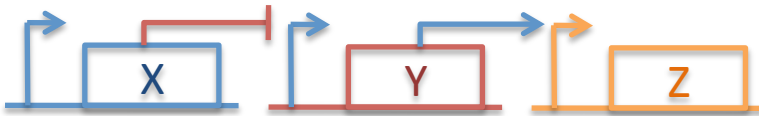
- Composition should preserve digital behavior



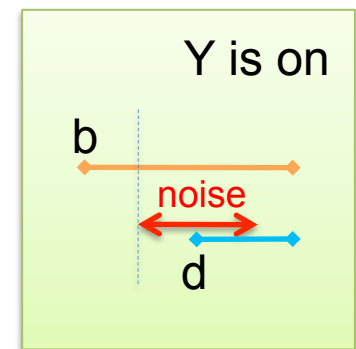
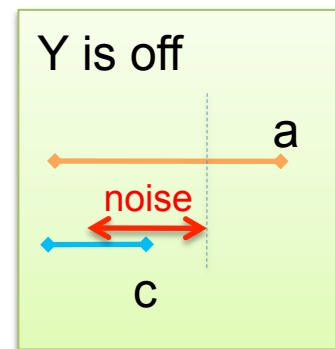
Signal Matching Problem: How do we pick the parts that have compatible interpretations for on/off so that when composed will preserve digital behavior?

Solution

- Pick the parts that are **signal compatible**
 - operate in same signal range where Signal = Concentration



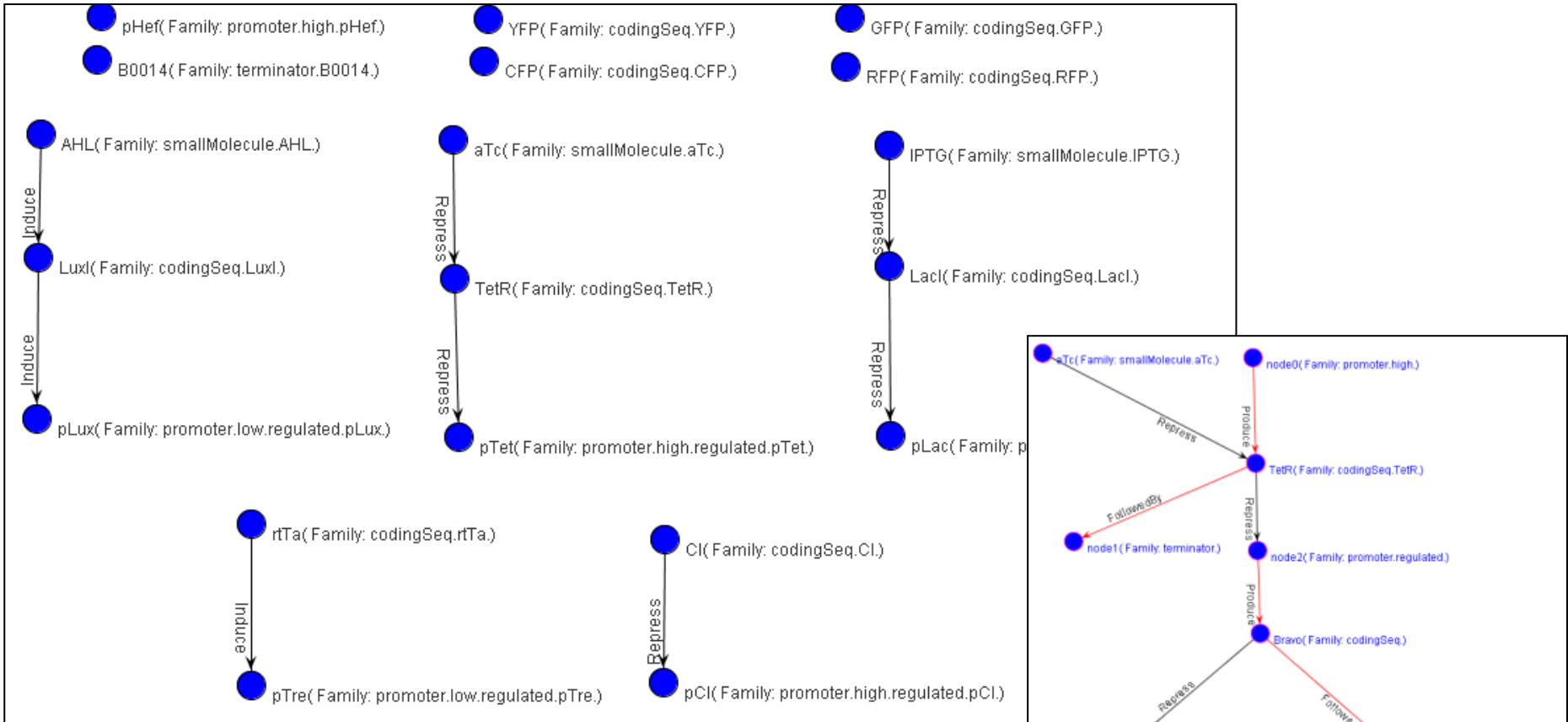
- Parts are signal compatible iff noisy output range is contained in valid input range



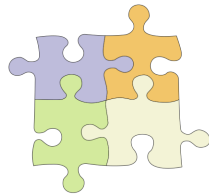
Feature Mapping + Signal Matching

- First do a feature mapping
 - Convert AGRN to GRN
- Check signal compatibility for every pair of bio-device in the GRN
 - If fails go back to previous step, find another GRN

Implementation: MatchMaker



www.clothocad.org

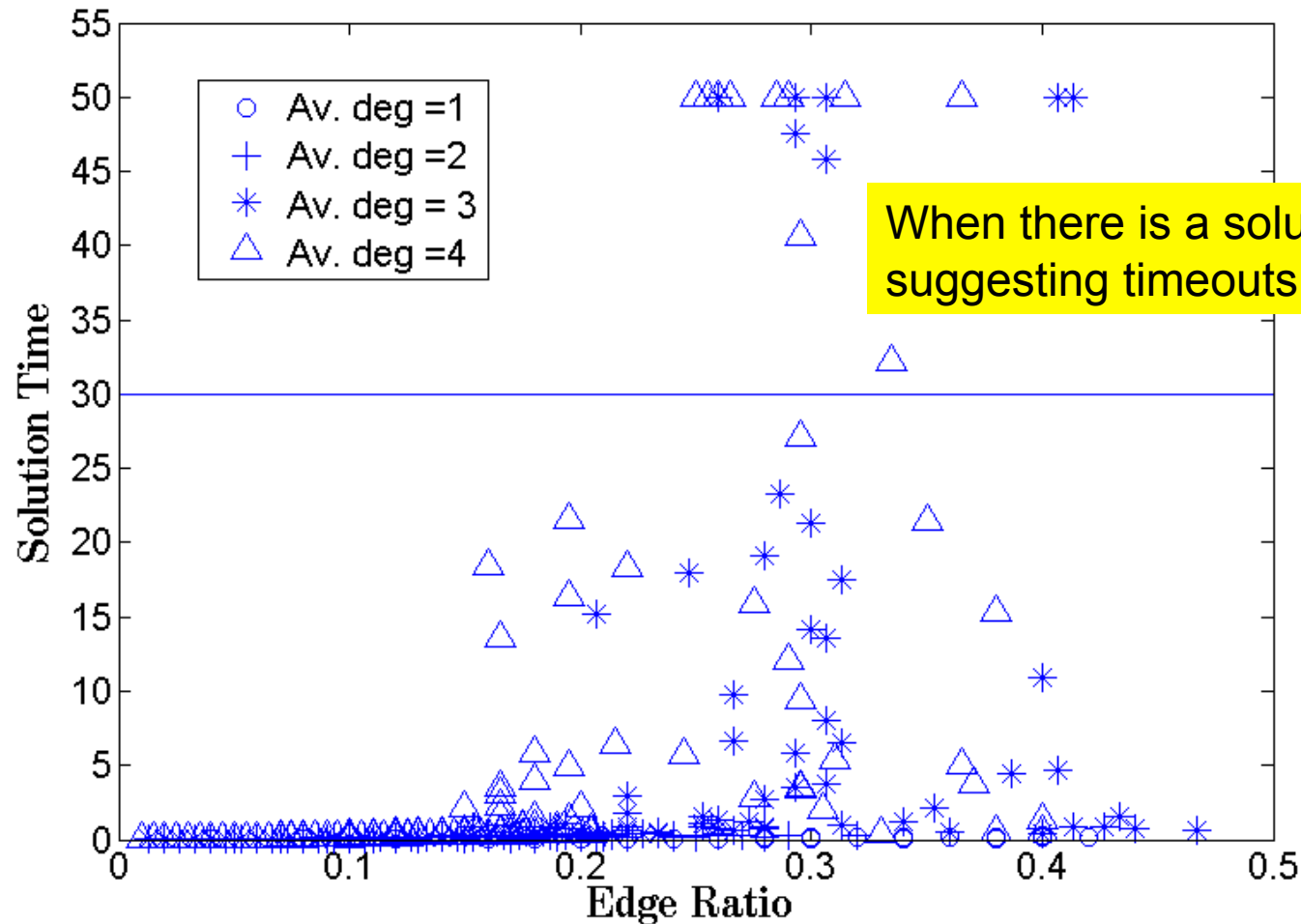


“MatchMaker” App

casbeer@eam@bbn.com

Empirical Results

- Experiments with random feature graphs (up to 200 nodes) and AGRNs (up to 60 nodes)



Conclusions

- Contribution
 - Fill in the big gap for going from AGRNs to GRNs
- Future plans and on going work
 - Hierarchical feature mapping
 - Search over families of features instead of individuals
 - Finding the most noise-tolerant network
 - Greedy search over signal data

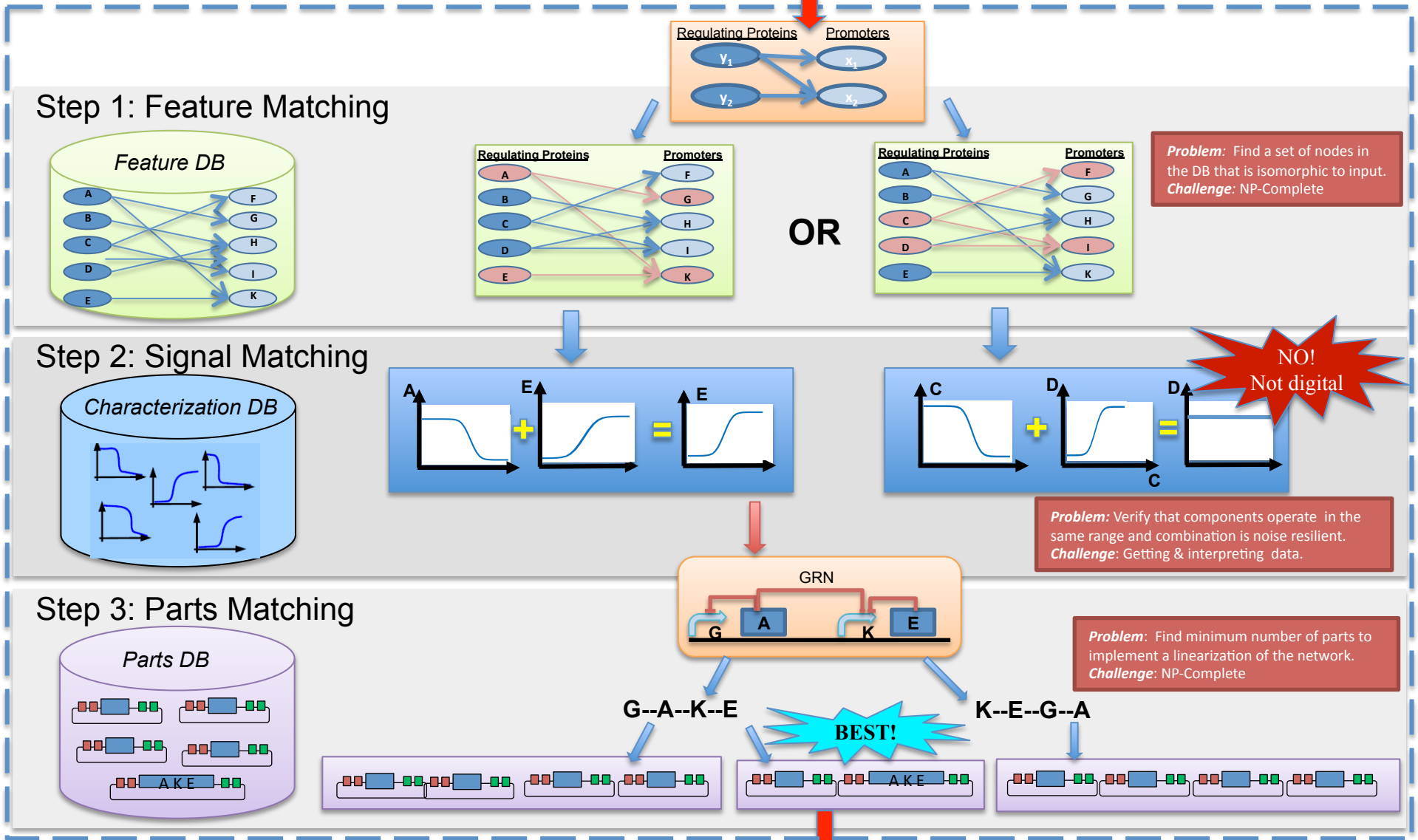
Questions?

BackUp Slides

Abstract GRN to Sequence of Parts

MatchMaker

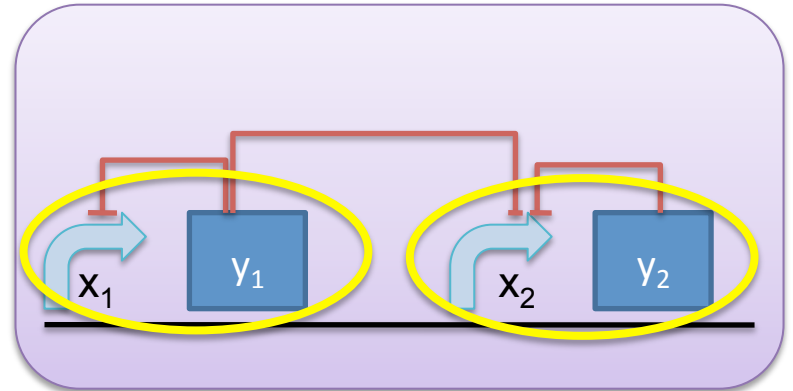
BioCompiler (previous step in tool-chain) output



tasbe-team@bbn.com
Input to the **Assembly Manager** (next step in the tool-chain)

Linearization

- Features in the A-GRN are loosely ordered
 - Y1 should be next to X1
 - Y2 should be next to X2
- Intuitively any total order that will satisfy these orderings is equivalently good.
 - Trivial solution: Linear time algorithm
 - $[X_1, Y_1, X_2, Y_2]$ or $[X_2, Y_2, X_1, Y_1]$
- Other design constraints may eventually affect orderings



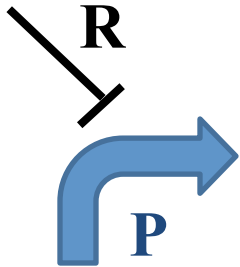
Part Mapping

- Basic: Given a sequence of features, what are the parts that can cover the sequence?
- Enhanced: Optimization problem
 - Minimize the number of parts used
 - Maximize the use of existing samples
- Current implementation addresses basic problem
 - Greedy search with preference on larger parts.

Next Steps

- Implementation of Families in Clotho to support hierarchical feature matching
- Multidimensional signal matching
- Optimal parts matching algorithm

Truth Table for a Hybrid Promoter?



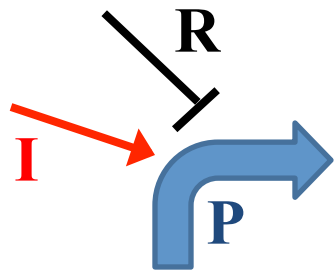
R	P_OUT
0	1
1	0

- Implicit Assumption:
P is constitutively high



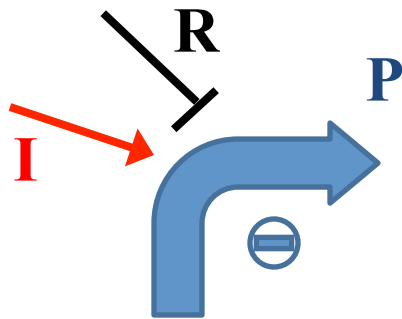
R	P_OUT
0	1
1	0

- Implicit Assumption:
P is constitutively low

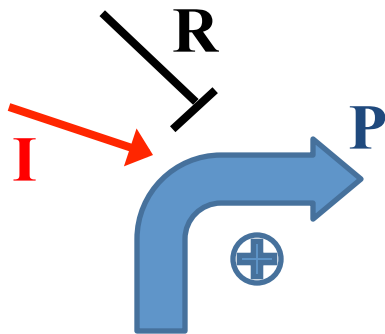


- Just seeing this symbol does not tell us what the intended behavior is
- Most of the time behavior depend on the type of the promoter and it should be part of the notation

Truth Table for a Hybrid Promoter?



R	I	P_OUT
0	0	0
0	1	1
1	0	0
1	1	0



R	I	P_OUT
0	0	1
0	1	1
1	0	0
1	1	1

Proposed Hierarchy of Promoters

