Cancer classification based on miRNA profiles using ASP

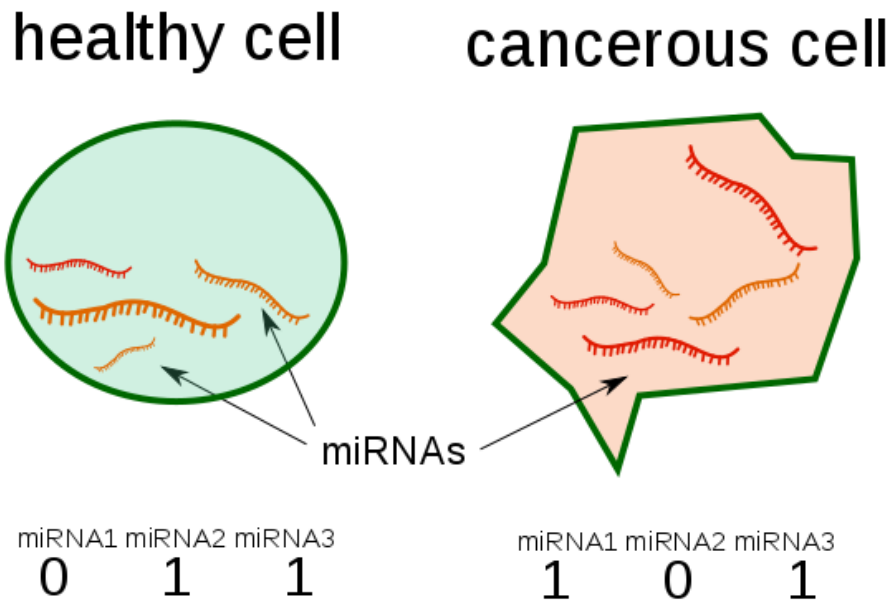
K. Becker and H. Klarner

Freie Universität Berlin, Germany

Berlin, Mai 2016

Selective cell targeting

- **Problem:** Discrimination of tumor from healthy tissues



- **Idea:** Cells differ in miRNA profiles
  $\rightarrow$ **in vitro classification using biochemical circuits**

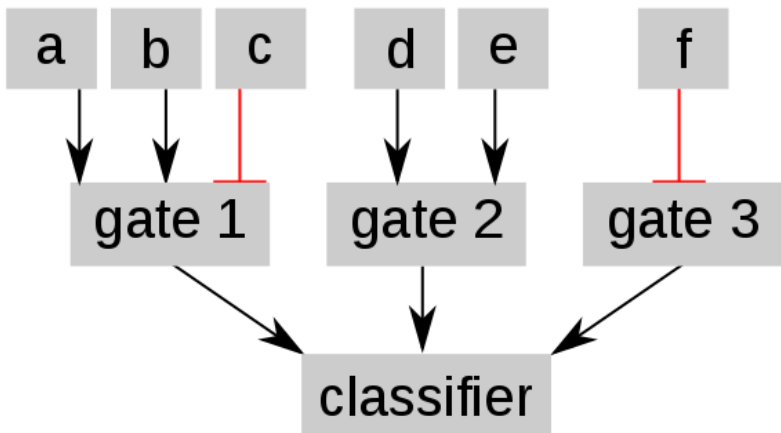
A boolean expression in conjunctive normal form (CNF)

- conjunction (AND) of *gates*
- each gate is a disjunction (OR) of literals
- **example:**

$$(a + b + !c) * (d + e) * (!f)$$

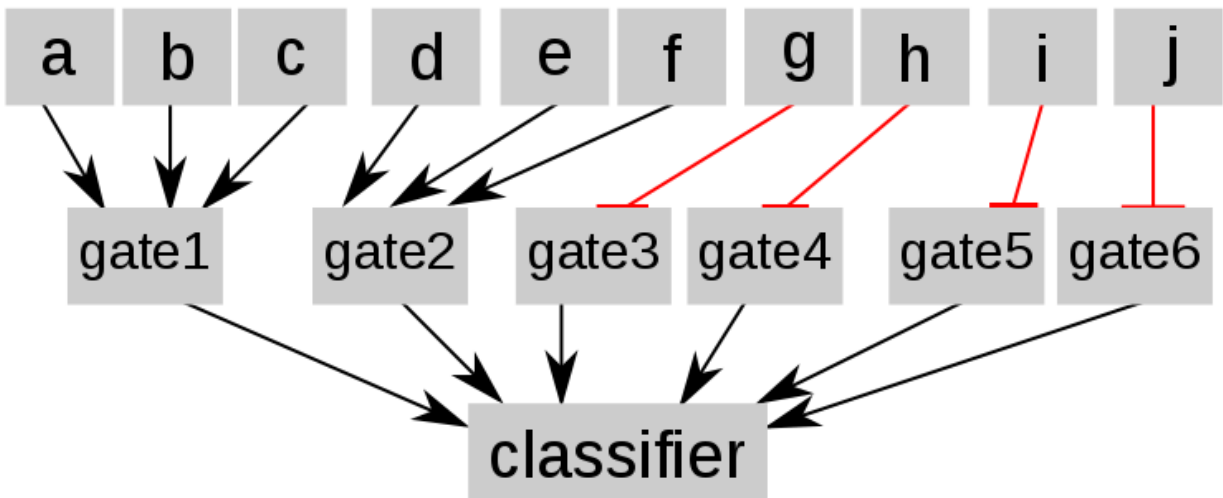
$+$ means disjunction, $*$ means conjunction, ! means negation

- CNF evaluates to 1 (*predicts cancer*) iff every gate evaluates to 1



Constraints from biology

- less than 10 inputs in total
- no more than 6 inputs attached to the AND gate
- no more than 3 inputs atttached to any OR gate
- no NOT gates attached to an OR gate
- no more than 2 OR gates
- no more than 4 NOT gates



Our encoding in ASP - Reminder

- facts:

```
P(A).
```

- conditional constraints:

```
Q(A) :- P(A).
```

- count constraints:

```
X { Q(A,B) } Y.
```

- conditional count constraints:

```
X { Q(A,B) } Y :- P(A).
```

- integrity constraints:

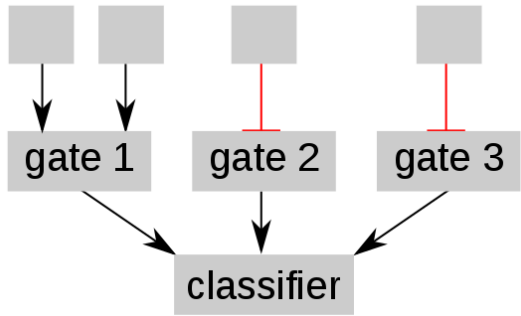
```
:- P(A).
```

Input: Data

		cancer?	**miRNAs**		
	ID	Annots	g1	g2	g3
tissues	1	0	1	1	0
	2	0	0	0	1
	3	1	0	1	0

```
tissue(1,healthy). tissue(2,healthy). tissue(3,cancer).

data(1,g1,high). data(1,g2,high). data(1,g3,low).
data(2,g1,low).  data(2,g2,low).  data(2,g3,high).
data(3,g1,low).  data(3,g2,high). data(3,g3,low).

is_mirna(Y) :- data(X,Y,Z).
```

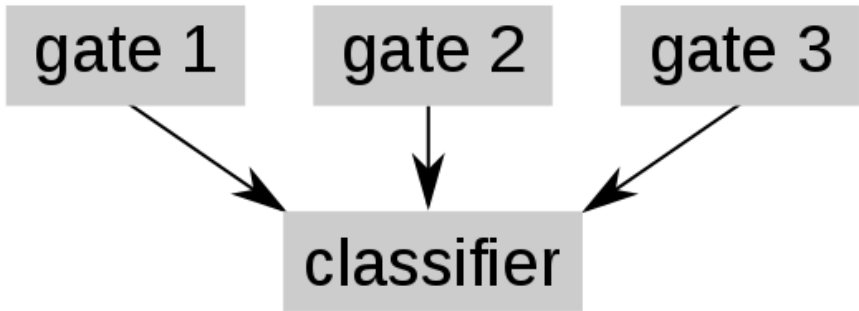
Input: Classifier structure




```
is_gate_type(1..2).

upper_bound_pos_inputs(type1, 2).
upper_bound_neg_inputs(type1, 0).
lower_bound_pos_inputs(type1, 0).
lower_bound_neg_inputs(type1, 0).
upper_bound_gate_occurence(type1, 1).

upper_bound_pos_inputs(type2, 0).
upper_bound_neg_inputs(type2, 1).
lower_bound_pos_inputs(type2, 0).
lower_bound_neg_inputs(type2, 0).
upper_bound_gate_occurence(type2, 2).

upper_bound_total_inputs(4).
```

Decision 1: How many gates do we use?

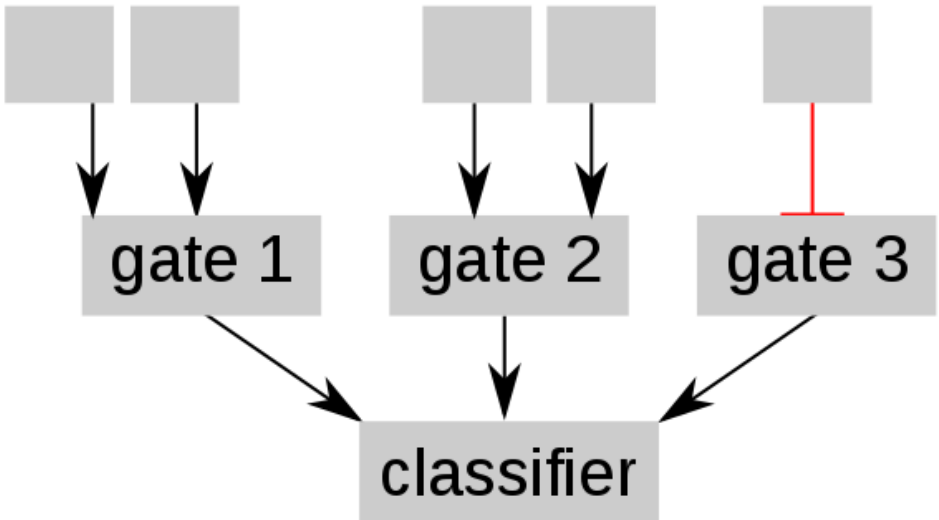
```
number_of_gates(3).

is_gate_id(1).
is_gate_id(2).
is_gate_id(3).
```




```
1 {number_of_gates(1..3)} 1.

is_integer(1..3).
is_gate_id(GateID) :- number_of_gates(X),is_integer(GateID),
                      GateID<=X.
```

Decision 2: What is the gate type of each gate?

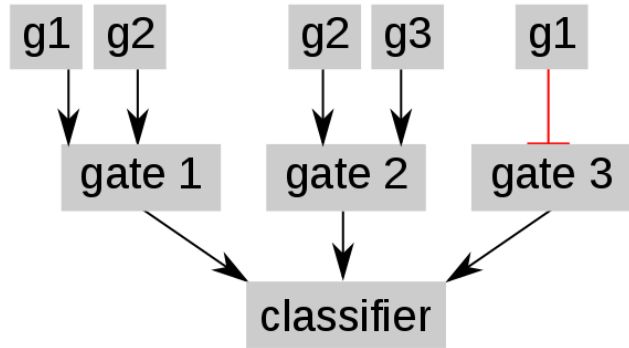
```
gate_type(1, type1).
gate_type(2, type1).
gate_type(3, type2).
```




```
1 {gate_type(GateID, X) : is_gate_type(X)} 1 :- is_gate_id(GateID).
```

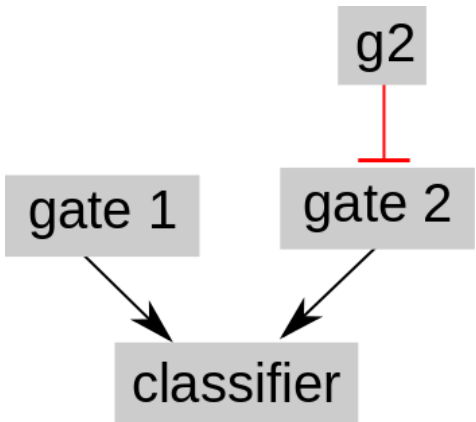
Decision 3: What are the inputs for our gates?

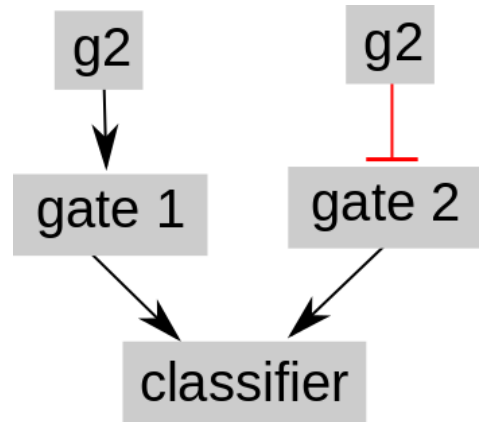
```
gate_input(1,positive,g1).
gate_input(1,positive,g2).

gate_input(2,positive,g2).
gate_input(2,positive,g3).

gate_input(3,negative,g1).
```




```
X {gate_input(GateID,positive,MiRNA):is_mirna(MiRNA)} Y
   :-gate_type(GateID,GateType),
     lower_bound_pos_inputs(GateType,X),
     upper_bound_pos_inputs(GateType,Y).

X {gate_input(GateID,negative,MiRNA):is_mirna(MiRNA)} Y
   :-gate_type(GateID,GateType),
     lower_bound_neg_inputs(GateType,X),
     upper_bound_neg_inputs(GateType,Y).
```

Gates must have inputs



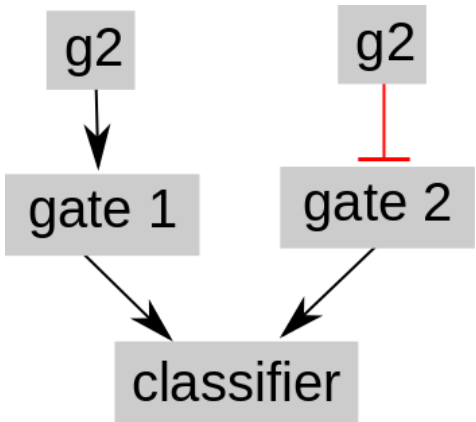


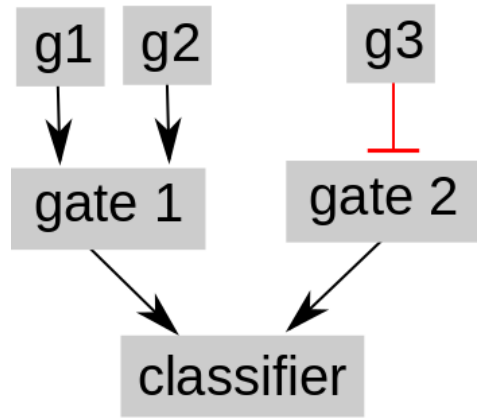

```
is_gate_id(1).
is_gate_id(2).
gate_input(2,negative,g2).
```

```
is_gate_id(1).
is_gate_id(2).
gate_input(1,positive,g2).
gate_input(2,negative,g2).
```

```
1 {gate_input(GateID, Sign, MiRNA):
 is_sign(Sign), is_mirna(MiRNA)} :- is_gate_id(GateID).
```

Inputs must be unique



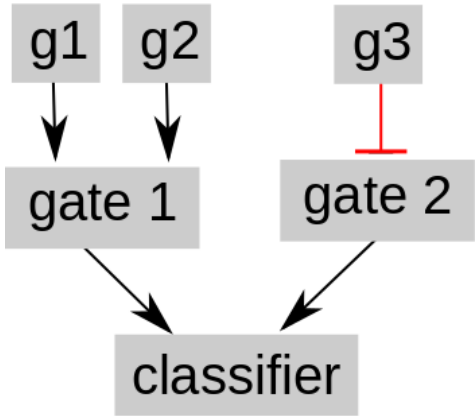



```
is_mirna(g1). is_mirna(g2).
is_mirna(g3).
gate_input(1,positive,g2).
gate_input(2,negative,g2).
```

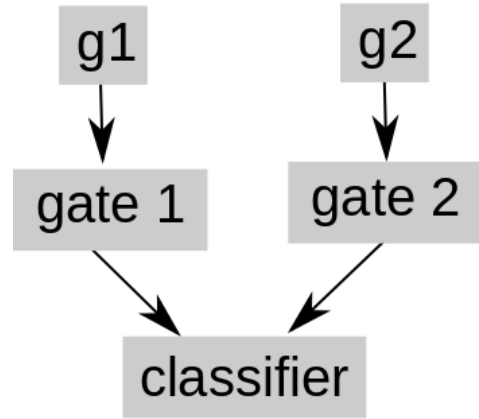
```
is_mirna(g1). is_mirna(g2).
is_mirna(g3).
gate_input(1,positive,g1).
gate_input(2,positive,g2).
gate_input(2,negative,g3).
```

```
{gate_input(GateID,Sign,MiRNA):
 is_sign(Sign), is_gate(GateID)} 1 :- is_mirna(MiRNA).
```

Number of inputs is bounded



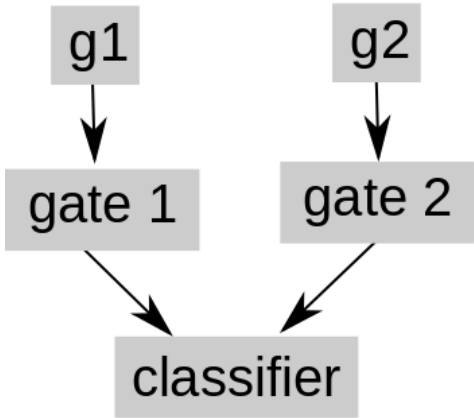

```
upper_bound_inputs(2).
gate_type(1,type1).
gate_type(2,type2).
gate_input(1,positive,g1).
gate_input(2,positive,g2).
gate_input(2,negative,g3).
```

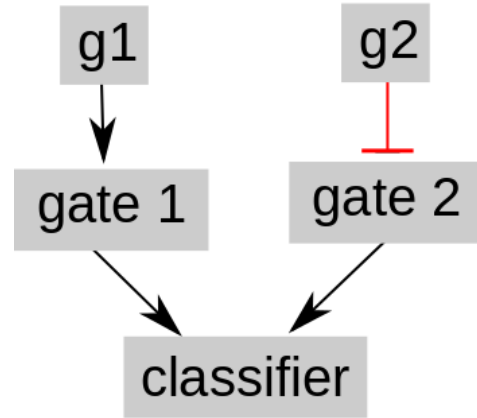



```
upper_bound_inputs(2).
gate_type(1,type1).
gate_type(2,type1).
gate_input(1,positive,g1).
gate_input(2,positive,g2).
```

```
{gate_input(GateID,Sign,MiRNA):
 is_gate_id(GateID), is_sign(Sign), is_mirna(MiRNA)} X :-
 upper_bound_inputs(X).
```

Occurences of gates



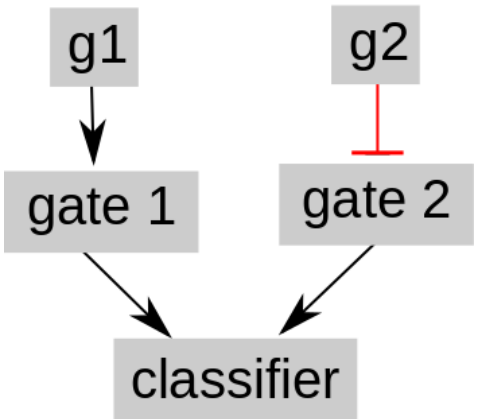



```
upper_bound_gate_occurence(type1, 1).
gate_type(1,type1).
gate_type(2,type1).
gate_input(1,positive,g1).
gate_input(2,positive,g2).
```

```
upper_bound_gate_occurence(type1, 1).
gate_type(1,type1).
gate_type(2,type2).
gate_input(1,positive,g1).
gate_input(2,negative,g2).
```

```
{gate_type(GateID,GateType): is_gate_id(GateID)} X :-
 upper_bound_gate_occurence(GateType,X).
```

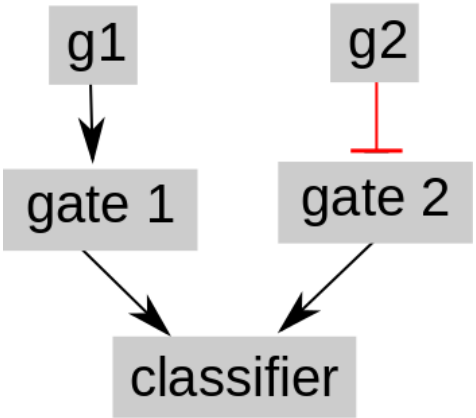
Consistency of gates with data



ID	Annots	g1	g2	g3	gate1	gate2
1	0	1	1	0	1	0
2	0	0	0	1	0	1
3	1	0	1	0	0	0

```
gate_fires(GateID,TissueID) :-
 gate_input(GateID,positive,MiRNA),
 data(TissueID,MiRNA,high).
```

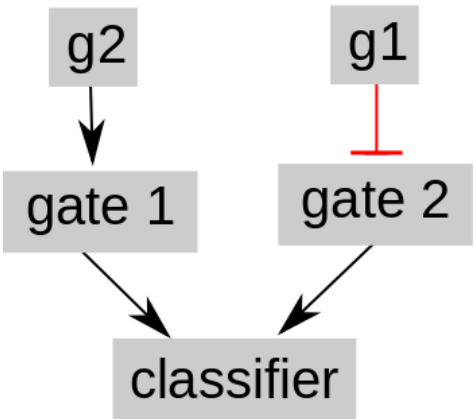
Consistency of classifier with gates



ID	Annots	g1	g2	g3	gate1	gate2	classifier
1	0	1	1	0	1	0	healthy
2	0	0	0	1	0	1	healthy
3	1	0	1	0	0	0	healthy

```
classifier(TissueID, healthy) :-
 not gate_fires(GateID, TissueID),
 is_gate_id(GateID), is_tissue_id(TissueID).

classifier(TissueID, cancer) :-
 not classifier(TissueID, healthy),
 is_tissue_id(TissueID).
```

Consistency of classifier with data



ID	Annots	g1	g2	g3	gate1	gate2	classifier
1	0	1	1	0	1	0	healthy
2	0	0	0	1	0	1	healthy
3	1	0	1	0	1	1	cancer

```
gate_input(1,positive,g2).
gate_input(2,negative,g1).
```

```
:- tissue(TissueID,healthy), classifier(TissueID,cancer).
:- tissue(TissueID,cancer),  classifier(TissueID,healthy).
```

Optimization

- single objective

```
#minimize{ 1,GateID:gate_input(GateID,Sign,MiRNA) }.
```

- with priorities

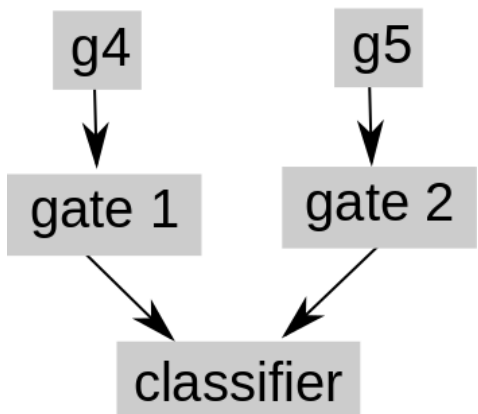
```
#minimize{ 1@1,GateID:gate_input(GateID,Sign,MiRNA) }.
#minimize{ 1@2,MiRNA: gate_input(GateID,Sign,MiRNA) }.
```

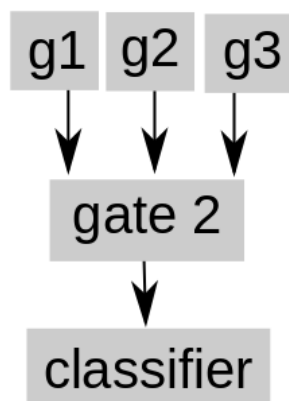
- weighted sum

```
#minimize{ 1@1,GateID:gate_input(GateID,Sign,MiRNA) }.
#minimize{ 8@1,MiRNA: gate_input(GateID,Sign,MiRNA) }.
```

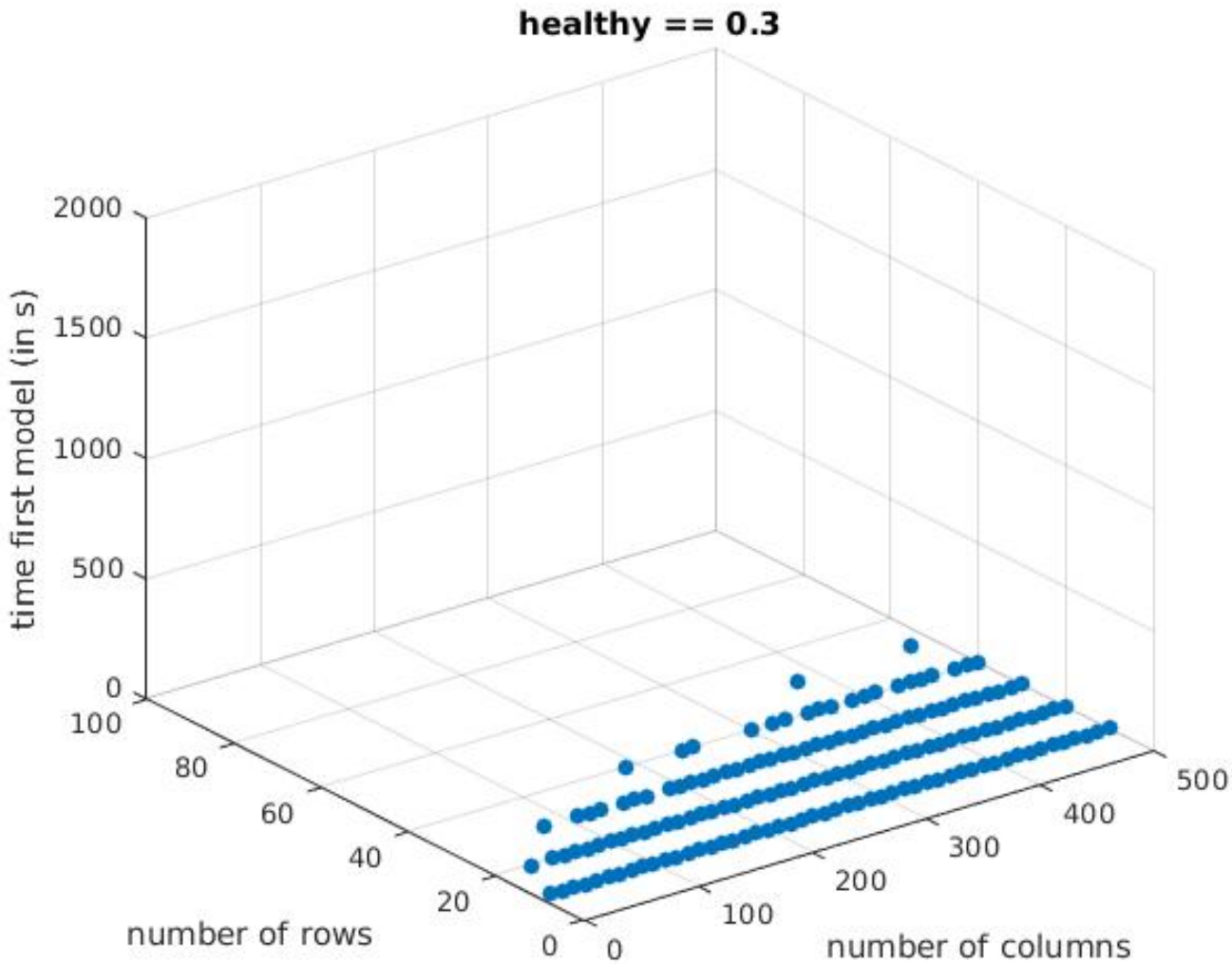
Priorities matter

ID	Annots	g1	g2	g3	g4	g5
1	1	0	0	1	1	1
2	1	0	1	0	1	1
3	1	0	1	1	1	1
4	1	1	0	0	1	1
5	1	1	0	1	1	1
6	1	1	1	0	1	1
7	1	1	1	1	1	1
8	0	0	0	0	0	1
9	0	0	0	0	1	0
10	0	0	0	0	0	0

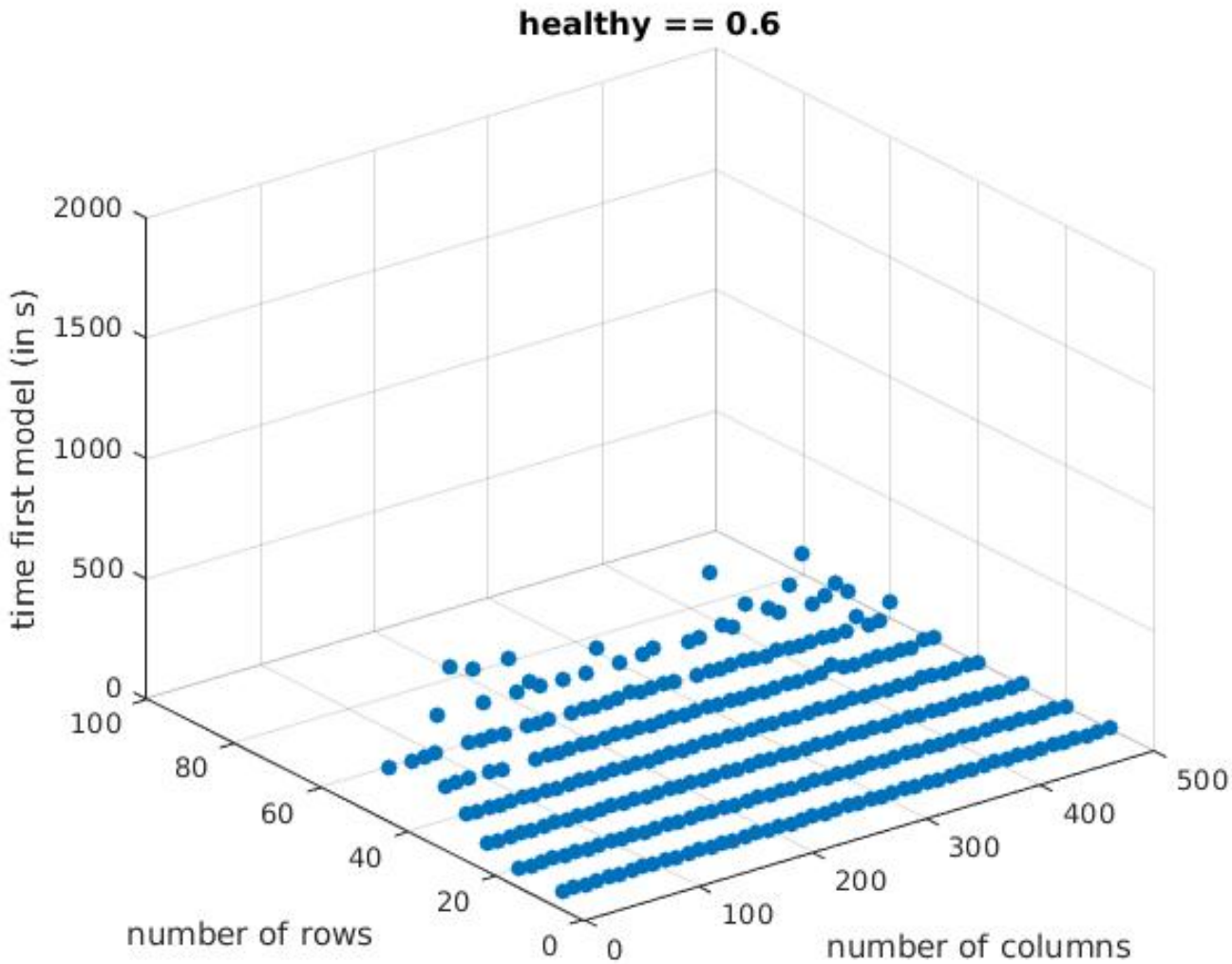




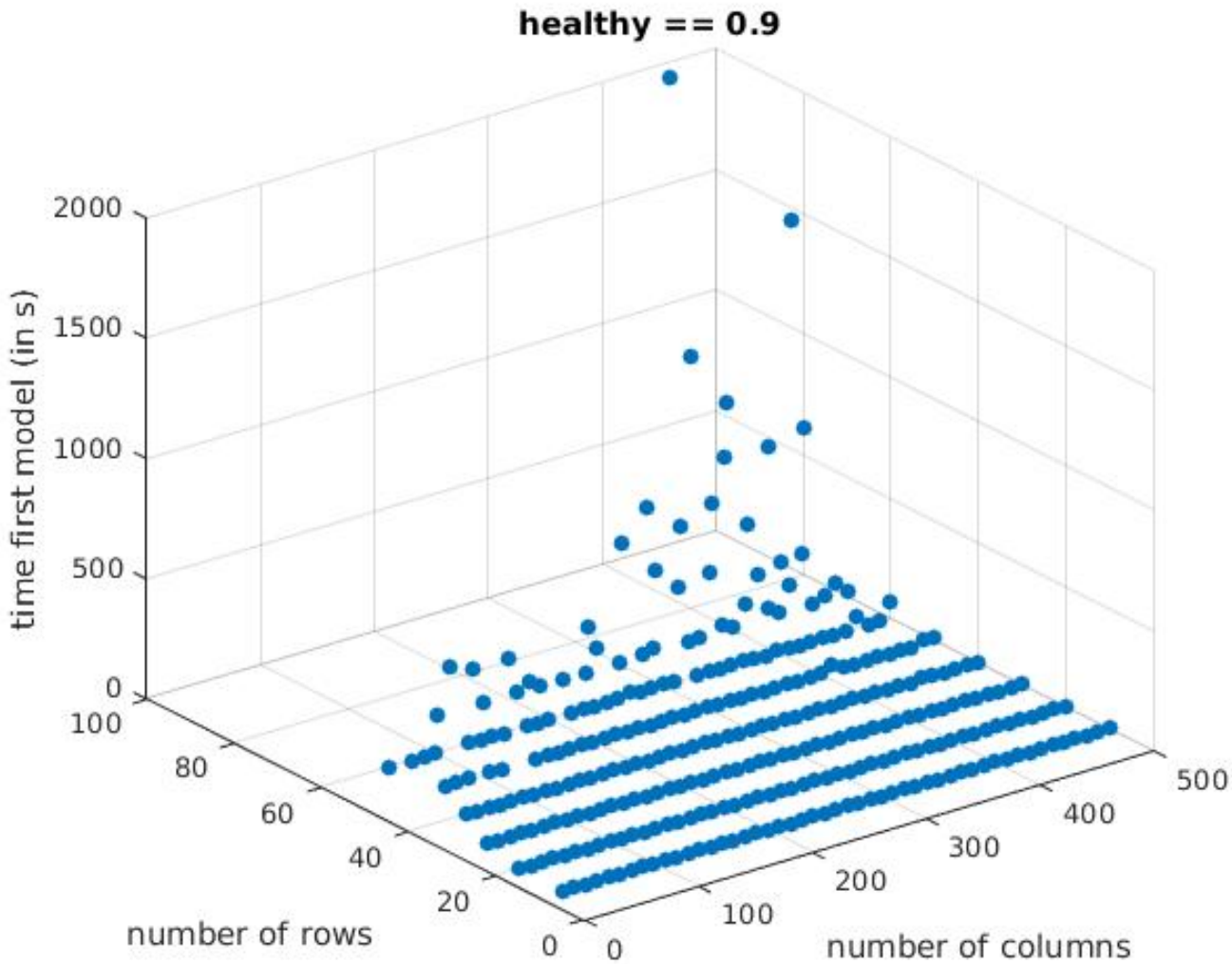
Time for generating first model



Time for generating first model



Time for generating first model



Summary

- enumerate Boolean expressions that agree with partial truth table
- biologically motivated constraints:
  - expression is in CNF
  - bounds on number of inputs and gates
  - definition of gates types with bounds on positive and negative inputs and on occurrences
- optimization by single objective, priorities and weighted sums

Breaking Symmetries?

```
Answer: 1
gate_input(1,negative,g3) gate_input(2,negative,g1)

Answer: 2
gate_input(1,positive,g2) gate_input(2,negative,g1)

Answer: 3
gate_input(2,negative,g3) gate_input(1,negative,g1)

Answer: 4
gate_input(2,positive,g2) gate_input(1,negative,g1)
```