Goal: Develop techniques that can be used to analyze models where the state space \((2^N)\) nodes) is large.

First, consider a more refined model for the lac operon.

*Added variables:*  
A. lac permease  
B. β-galactosidase  
A. allolactose  
B. repressor protein lacI  
C. catabolite activator protein CAP

*New Feature:* Distinguish between 3 levels of lactose & allolactose:  
none, low, high.

This is in contrast to proteins & enzymes, which are either in abundance, or absent (concentration levels when expressed are thousands of times lower than when they're not.)

Several ways to do this:

1. Use states \(S = \{0, 1, 2\} \) instead of \(\{0, 1\}\)
   
   0 = none  
   1 = low  
   2 = high
(ii) Introduce additional variables: \( L_e \) and \( A_e \) that represent "low levels."

- No lactose: \( L_e = 0, \ L = 0 \)
- Low lactose: \( L_e = 1, \ L = 0 \)
- High lactose: \( L_e = 1, \ L = 1 \)

The other possibility, \( L_e = 0, \ L = 1 \) is meaningless; ignore it.

Assumption: High levels of lactose or allolactose at time \( t \) means at least low levels at time \( t+1 \).

**Proposed model:**

\[
\begin{align*}
F_M &= \overline{R} \land C \quad \text{"no repressor protein \( \xi \), high conc. of CAP"} \\
F_P &= M \\
F_C &= \overline{G_e} \\
F_A &= L \land B \\
F_L &= \overline{G_e} \land P \land L_e \\
F_B &= M \\
F_R &= \overline{A} \land \overline{A_e} \quad \text{"no allolactose"} \\
F_{A_e} &= A \lor L \lor L_e \\
F_{L_e} &= \overline{G_e} \land (L \lor L_e)
\end{align*}
\]

*Wiring diagram*
Problem: State space has size $2^9 = 512$ nodes.

This is manageable (barely).

But some Boolean networks are too big.

Ex: A model for T cell receptor signaling contains 94 nodes, so $2 \cdot 10^{28}$ nodes in the state space.

Goal: How do we find the fixed points easily?

Recall: A fixed point $(p_1, p_2, \ldots, p_n)$ satisfies

$$p_i = F_{x_i}(p_1, \ldots, p_n)$$
$$p_2 = F_{x_2}(p_1, \ldots, p_n)$$
$$\vdots$$
$$p_n = F_{x_n}(p_1, \ldots, p_n)$$

Approach: Computational algebra (=abstract algebra) using algebraic geometry (=polynomial algebra) & Gröbner bases.

Problem rephrased: Find solutions to the system of polynomial equations:

$$\begin{cases}
F_{x_1}(x_1, \ldots, x_n) - x_1 = 0 \\
F_{x_2}(x_1, \ldots, x_n) - x_2 = 0 \\
\vdots \\
F_{x_n}(x_1, \ldots, x_n) - x_n = 0
\end{cases}$$
Step 1: Write functions in polynomial form.

Recall
\[ x_1 \land x_2 = x_1 x_2 \]
\[ x_1 \lor x_2 = x_1 + x_2 + x_1 x_2 \]
\[ \overline{x}_1 = x_1 + 1 \]

Ex: \[ F_{a_2} = A \lor L \lor L_2 = (A \lor L) \lor L_2 \]
\[ = (A + L + AL) \lor L_2 \]
\[ = A + L + AL + L_2 + (A + L + AL) L_2 \]
\[ = A + L + AL + L_2 + AL_2 + LL_2 + ALL_2 \]

Do this for remaining 8 functions (exercise).

Step 2: Use computational algebra software to find a Gröbner basis of these polynomials.

Gröbner bases are a generalization of Gaussian elimination, but for systems of polynomials instead of linear equations.

Gaussian elimination: Input linear system.

\[ \begin{align*}
\text{e.g.,} \quad & \begin{cases} x + 2y = 1 \\ 3x + 8y = 1 \end{cases}
\end{align*} \]

\[ \begin{bmatrix} 1 & 2 & 1 \\ 3 & 8 & 1 \end{bmatrix} \sim \begin{bmatrix} 1 & 2 & 1 \\ 0 & 2 & -2 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & -1 \end{bmatrix} \]

Output: A simpler system e.g., \[ \begin{cases} x = 3 \\ y = -1 \end{cases} \] with the same solutions!
Remark: If the output system is "uppertriangular", then we can back-substitute \$x\$, solve completely. e.g., \[
\begin{align*}
x + z &= 2 \\
y - z &= 8 \\
o &= 0
\end{align*}
\]

Gröbner bases give us a much simpler set of polynomials that have the same solution set as the original system.

The theory is deep, but it's implemented in software packages.

*Free open source package: SAGE*  
[www.sagemath.org](http://www.sagemath.org)

Select "Try Sage online" to create a notebook account. (sagenb.org)

Example: let's solve the following system: \[
\begin{align*}
x^2 + y^2 + z^2 - 1 &= 0 \\
x^2 + z^2 - y &= 0 \\
x - z &= 0
\end{align*}
\]

Enter in SAGE (hit Shift+Return after each line)

```
P.<x,y,z> = PolynomialRing(RR, 3, order='lex'); P
I = ideal(x^2 + y^2 + z^2 - 1, x^2 + z^2 - y, x - z); I
B = I.groebner_basis(); B
```

**Output:** \[x - z, y - 2x^2 + 2, z^4 + \frac{1}{2}x^2 z^2 - \frac{1}{4}\]

That is, \[
\begin{align*}
x^4 + \frac{1}{2}x^2 z^2 - \frac{1}{4} &= 0 \\
y - 2x^2 &= 0 \\
x - z &= 0
\end{align*}
\]

Now back-substitute!
Note: This system is "upper triangular", i.e., we can
- Solve for \( z \) in Eq 1
- Plug into Eq 2 to solve for \( y \)
- Plug into Eq 3 to solve for \( x \)

Solutions:
\[
\begin{align*}
\text{Solve 1:} & \quad z = \pm \sqrt{-1 + \sqrt{5}} \quad ; \quad y = \pm z^2 \quad ; \quad x = \mp 2 \\
\text{Solve 2:} & \quad z = \sqrt{-1 + \sqrt{5}} \quad ; \quad y = \frac{1 + \sqrt{5}}{2} \quad ; \quad x = \frac{1 + \sqrt{5}}{2} \\
\text{and} & \quad z = \sqrt{-1 - \sqrt{5}} \quad ; \quad y = \frac{1 - \sqrt{5}}{2} \quad ; \quad x = \frac{1 - \sqrt{5}}{2}
\end{align*}
\]

lac operon example:
\[
\frac{f_c(x_1, \ldots, x_n)}{x_i} - x_i \quad \text{(recall: } \frac{x_i}{x_i} = 1) \]

\[
\begin{align*}
M & \quad x_1 \quad & x_1 + x_4 x_5 + x_4 = 0 \quad \text{(Put } L_e = q = 0) \\
P & \quad x_2 \quad & x_1 + x_2 = 0 \quad \text{(Put } G_e = g = 0) \\
N & \quad x_3 \quad & x_1 + x_3 = 0 \\
C & \quad x_4 \quad & x_4 + (g + 1) = 0 \\
R & \quad x_5 \quad & x_5 + x_6 x_7 + x_6 + x_7 + 1 = 0 \\
A & \quad x_6 \quad & x_6 + x_3 x_8 = 0 \\
\text{A.e} & \quad x_7 \quad & x_6 + x_7 + x_8 + x_9 + x_8 x_9 + x_8 x_9 + x_6 x_9 x_9 = 0 \\
L & \quad x_8 \quad & x_8 + a(g + 1) x_2 = 0 \\
L_e & \quad x_9 \quad & x_9 + (g + 1)(x_8 + a x_8 + a) = 0
\end{align*}
\]

Output (SAGE):
\[
[x_1, x_2, x_3, x_4+1, x_5+1, x_6, x_7, x_8, x_9]
\]
We have found the (unique) fixed point: when \( L_e = a = 0, \ G_e = g = 0 \)

\[
(M, P, b, C, P, A, A_e, L, L_e) = (x_1, x_2, ..., x_9) = (0, 0, 0, 1, 1, 0, 0, 0)
\]

**Exercise:**
- If \((L_e, G_e) = (0, g) = (0, 1)\), then the fixed point is
  \[
  (0, 0, 0, 0, 1, 0, 0, 0, 0)
  \]
- If \((L_e, G_e) = (a, g) = (1, 0)\), the fixed point is
  \[
  (0, 0, 0, 0, 1, 0, 0, 0, 0)
  \]
- If \((L_e, G_e) = (a, g) = (1, 1)\), the fixed point is
  \[
  (1, 1, 1, 0, 1, 1, 1, 1)
  \]

**Question:** Do these make biological sense? (They do!)

- Predicts operon is on only when external lactose is available and external glucose is not.

In these 3 cases, all variables (except repressor protein) are present.

- When glucose is available, operon is off.

**Remark:** We didn't need to analyze all \(2^9 = 512\) states. Most of them are irrelevant for us.