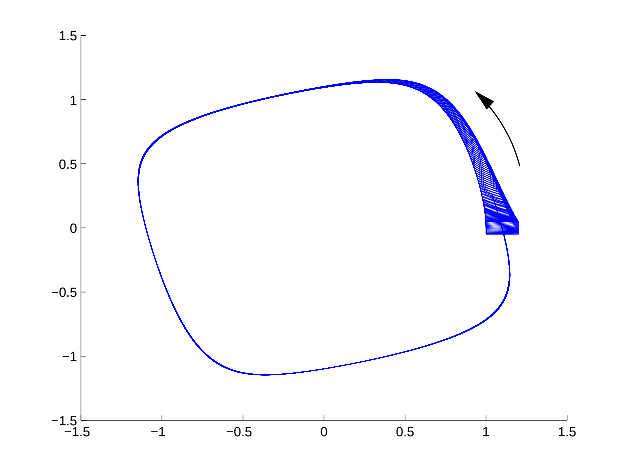
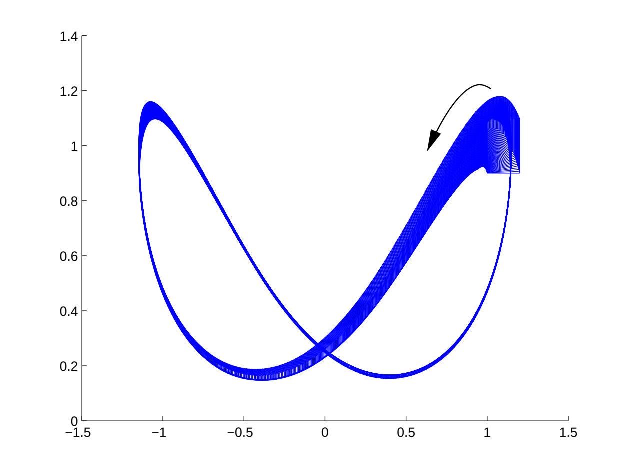
# Coho: A Verification Tool for Circuit Verification by Reachability Analysis

Chao Yan

Supervisor: Mark Greenstreet



# Overview

- Coho System
  - A verifi cation tool for systems modeled by nonlinear ordinary differential equations (ODEs).
  - Approximate the ODE model by linear differential inclusion.
  - Represents the reachable region as a projectagon.
  - Make extensive of linear programs.
- New Implementation of *Simplex* Based on Interval Arithmetic
  - Error bound is produced.
  - Ill-conditioned problem is easy to detect.
  - Modifi ed rule for pivot selection.
- O(n) Linear System Solver
- Projection Algorithm Using Linear Programming
- Coho Applications

# Circuit Verification

- Verifi cation of Circuit
  - Verify the circuit satisfi es high level specifi cation.
  - Simulation doesn't guarantee fully correctness.
  - Test coverage decreases with the increasing of circuit complexity.
- Formal Verifi cation
  - Proving using mathematical methods.
  - Industrial success for digital system.
- Digital Circuit are Really Analog
  - High speed, low power circuit.
  - New opportunities for verifi cation.

# Coho: Verification as Reachability

- Verifi cation is a $R^d$ Reachability Problem
  - Reachability analysis is to compute the reachable state according to the dynamics model.
  - Example: safety requirement.
- Circuit Modeling
  - Circuit is modled by ODEs:

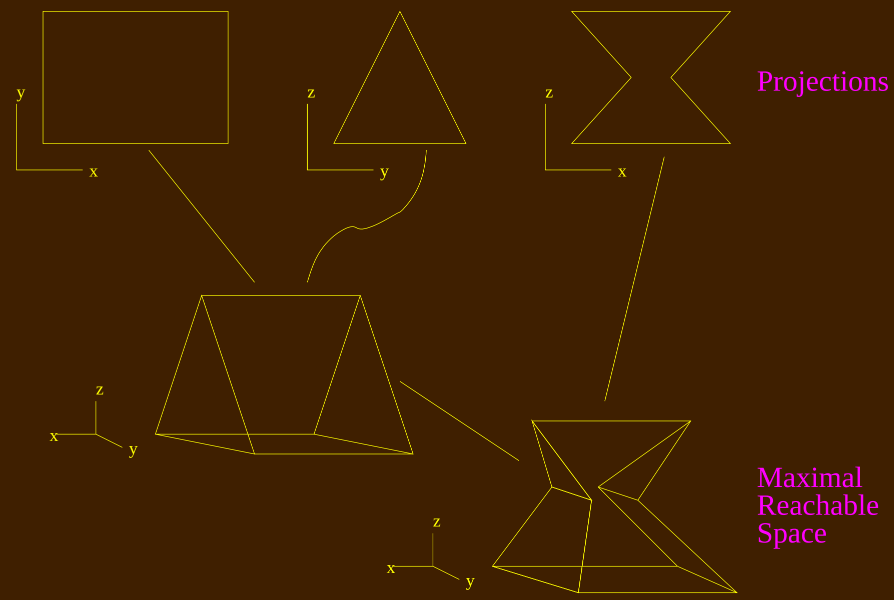
$$\dot{x} = F(x, t).$$

  - Specifi cation is a predictor of *safe* region.
- Approximation
  - Exact solution does not exist for general ODE.
  - Coho linearizes the model by differential inclusion:

$$F(x, t) \subseteq Ax + b + cube(u).$$

# Projectagon

- How to represent reachable region?
  - Effi ciency of geometry computation.
  - Approximation error of reachable region.
- Projectagon
  - Compact representation by its projection onto two dimensional subspace.
  - A edge corresponds to a face, which allows many operations on faces to be carried out as simple operations on edge.
  - Over approximated as required by verifi cation.



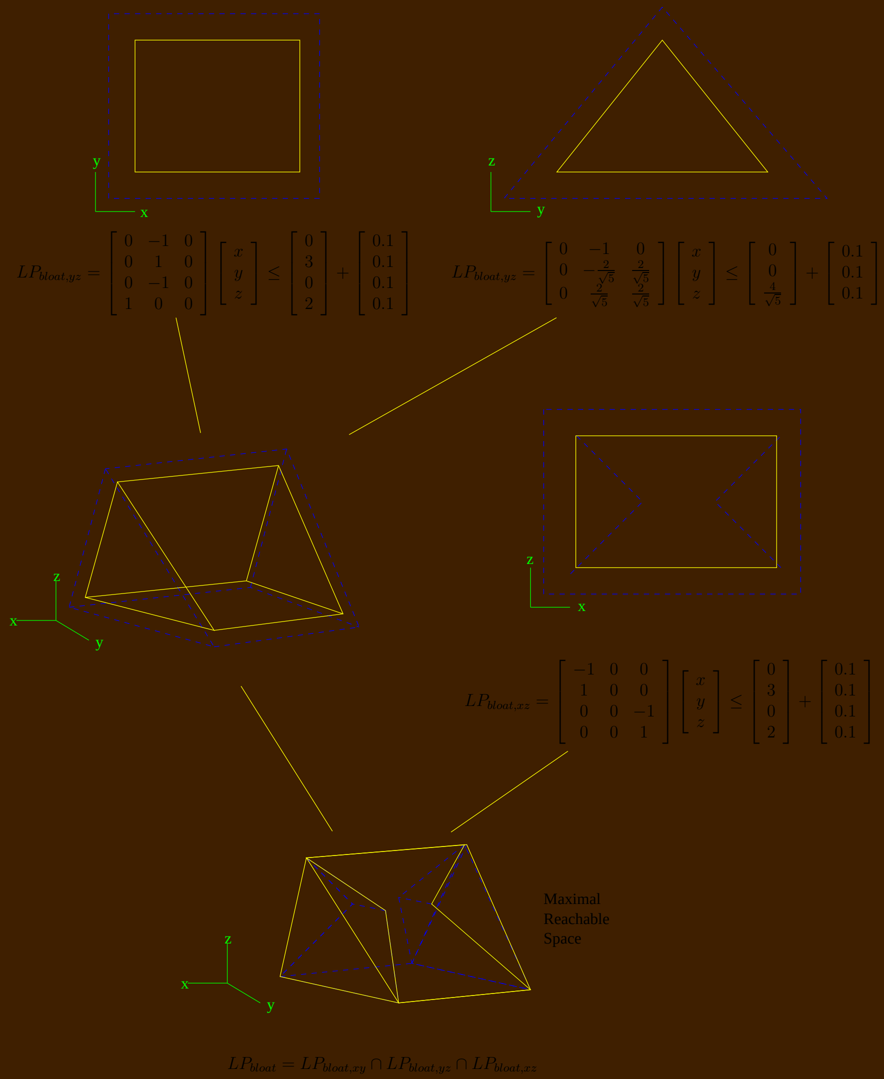
# Coho Linear Program

- Projectagon as Linear Program

$$
\begin{array}{lll}
 & \min & c^T x \\
s.t. & Ax & \geq b
\end{array}
$$

  where $x$ is a point of a projectagon.

- Special Structure of Coho Linear Program
  - Each constraint is an edge of projection polygon.
  - One or two non-zero elements for each row/constraint.



# Coho Linear Program Solver

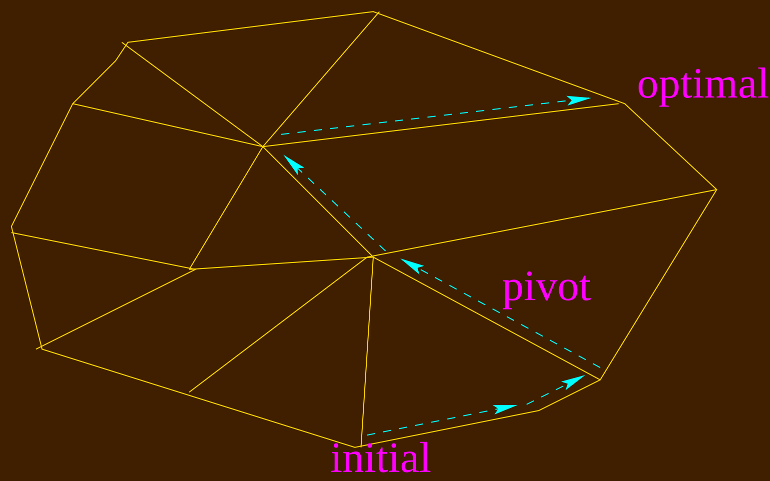
- Simplex
  - For standard form linear programs.

$$\begin{array}{rll} & \min & c^T x \\ s.t. & Ax & = b \\ & x & \geq 0 \end{array}$$

  - Find more favorable feasible basis by pivoting till the optimal one.
- Convert Coho Linear Program to Standard Form
  - Slack variables? does not preserves the special structure.
  - Dual of Coho linear program is standard and has the same optimal value.

# Coho Linear Program Solver

- Simplex
  - For standard form linear programs.
  - Find more favorable feasible basis by pivoting till the optimal one.




- Convert Coho Linear Program to Standard Form
  - Slack variables? does not preserves the special structure.
  - Dual of Coho linear program is standard and has the same optimal value.

# Coho Linear Program Solver

- Simplex
  - For standard form linear programs.
  - Find more favorable feasible basis by pivoting till the optimal one.
- Convert Coho Linear Program to Standard Form
  - Slack variables? does not preserves the special structure.
  - Dual of Coho linear program is standard and has the same optimal value.

$$\begin{array}{rcl} & \min & c^T x \\ s.t. & Ax & \geq b \end{array} \quad \Rightarrow \quad \begin{array}{rcl} & \min & -b^T u \\ s.t. & A^T u & = c \\ & u & \geq 0 \end{array}$$

# Coho Linear System Solver

- Tableau Computation
  - Simplex makes pivoting based on the tableau matrix.

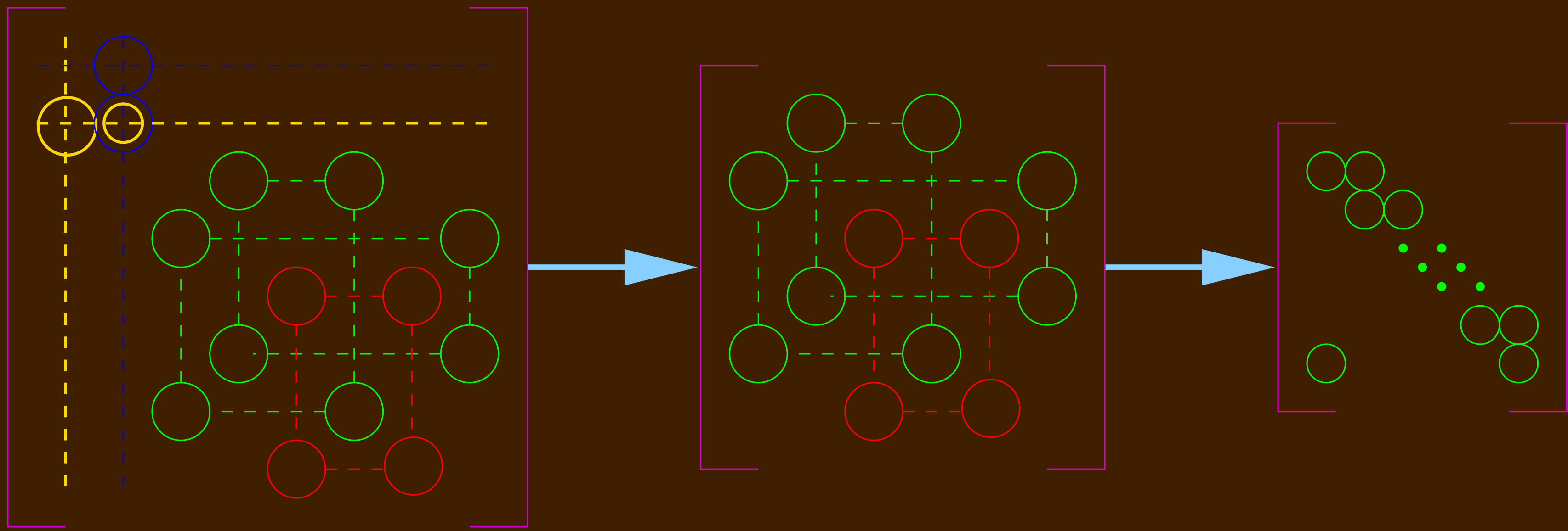
$$T = A_{\mathcal{B}}^{-1}[A|b], \quad \mathcal{B} \text{ is a basis}$$

  - Expensive computation vs. Accumulated error.
- Coho Linear System

$$Ax = y$$

  - One or two non-zero elements for each row/column of $A$.
  - Linear time algorithm to compute the solution.
  - Reduced accumulated error by computing tableau matrix directly from input data.

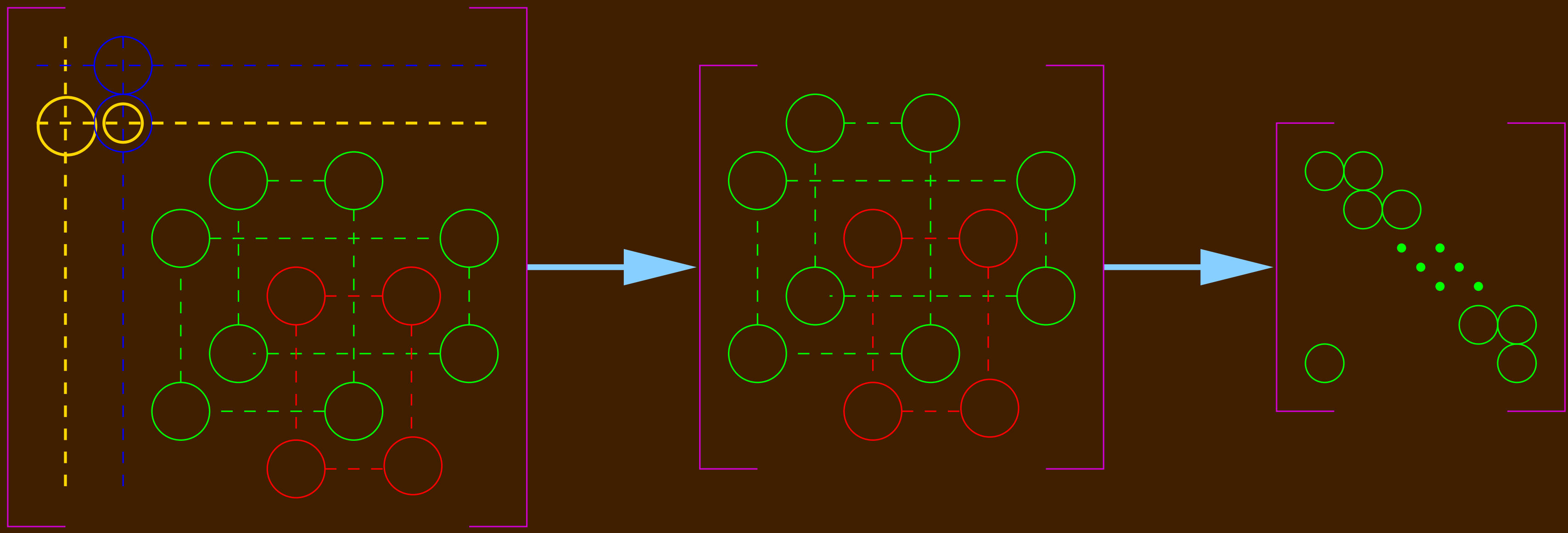
# Convert to Cycle Matrix



Eliminate singleton rows — Partition into cycles — Solve each cycle

- Solve singleton rows directly
- Eliminate the singleton columns
- Partition into cycles
- Solve each cycle
- Solve singleton columns

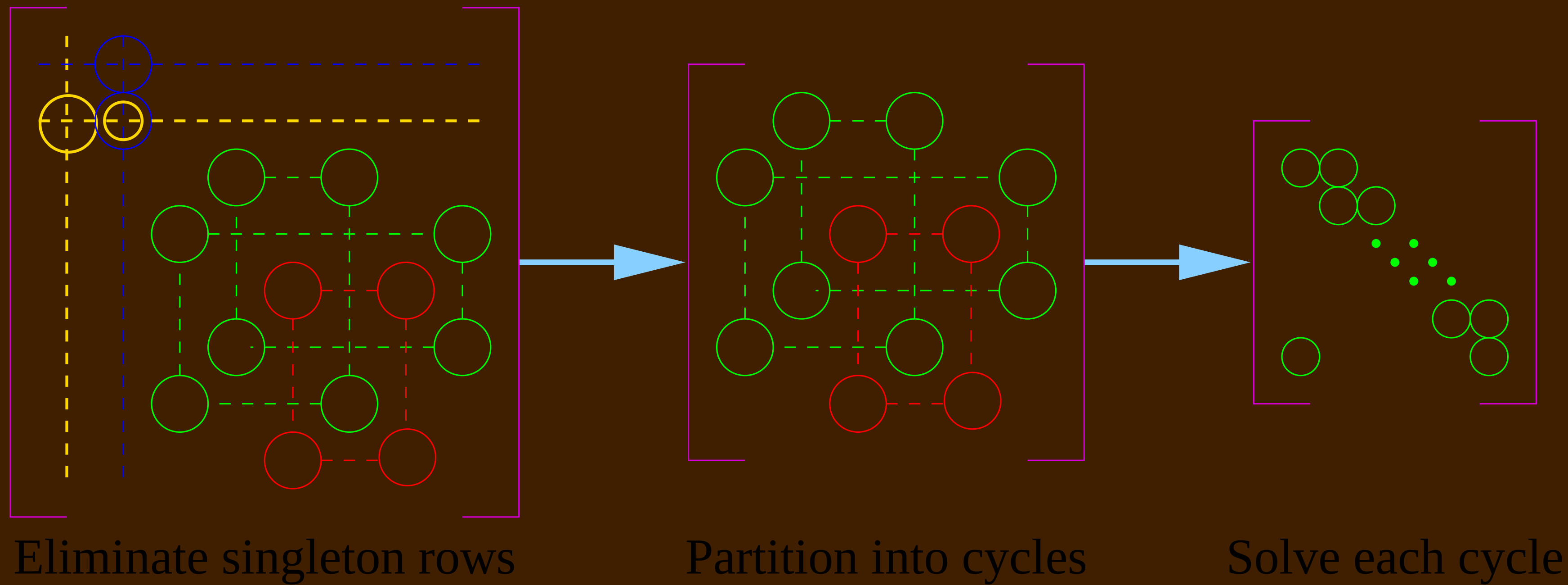
# Convert to Cycle Matrix



Eliminate singleton rows    Partition into cycles    Solve each cycle

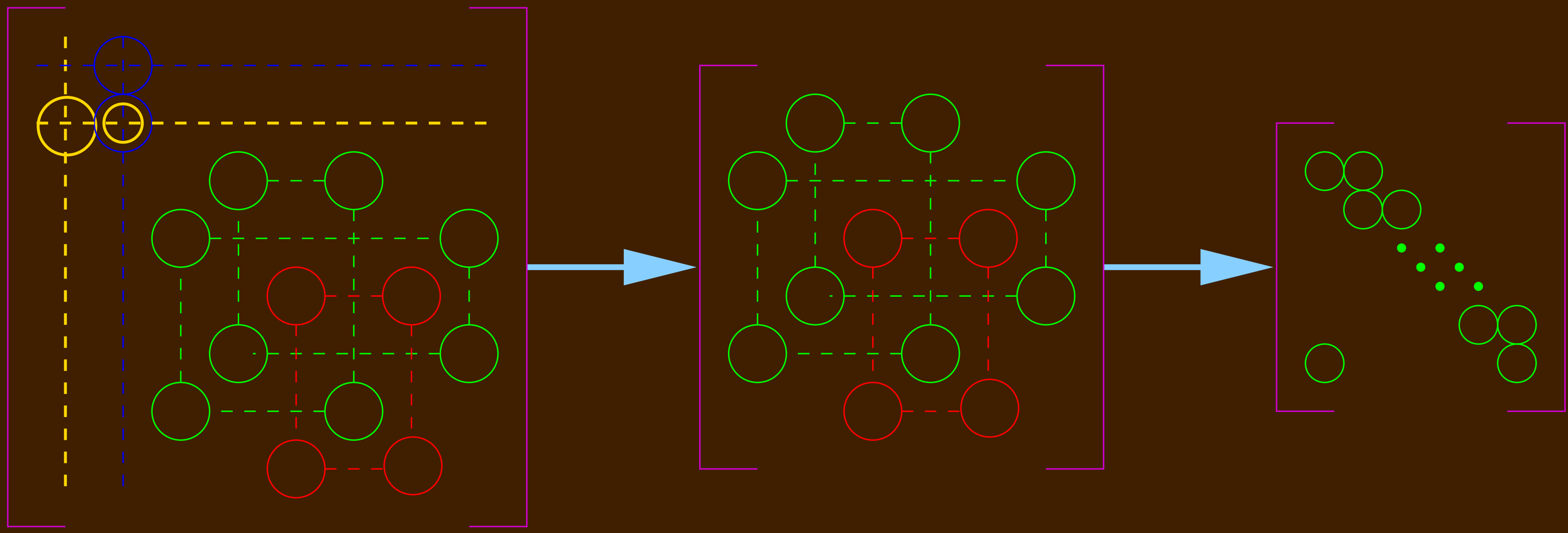
- Solve singleton rows directly
- Eliminate the singleton columns
- Partition into cycles
- Solve each cycle
- Solve singleton columns

# Convert to Cycle Matrix




- Solve singleton rows directly
- Eliminate the singleton columns
- Partition into cycles
- Solve each cycle
- Solve singleton columns

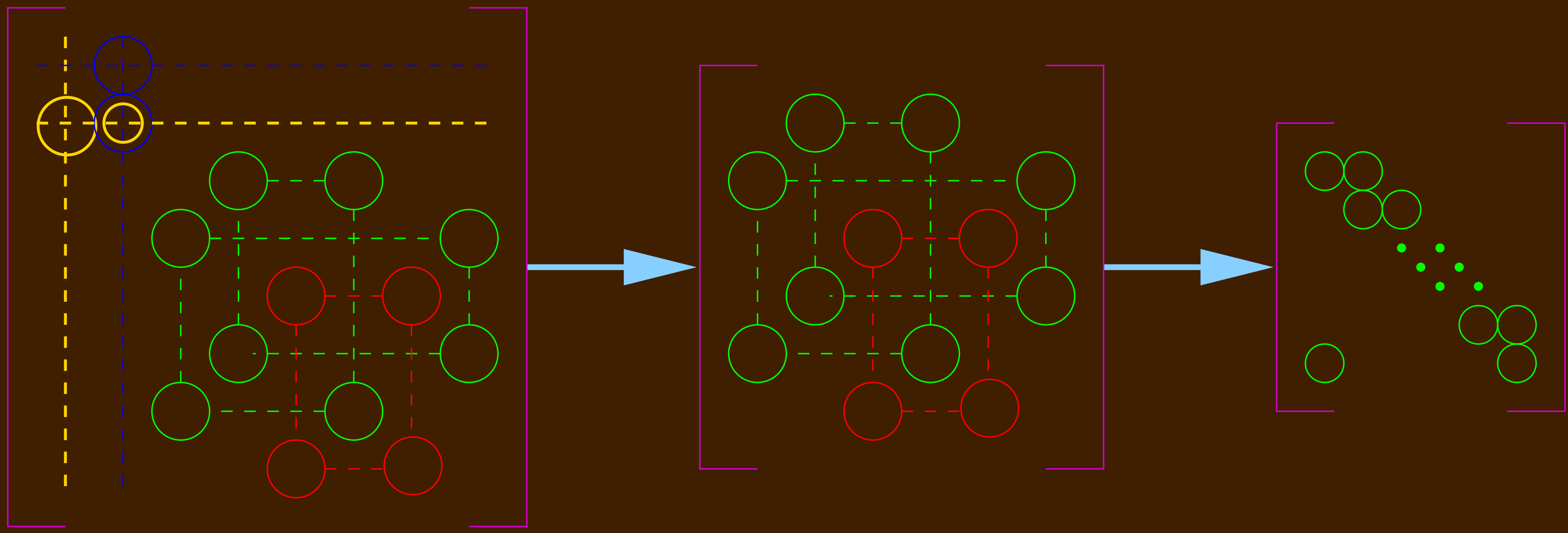
# Convert to Cycle Matrix



Eliminate singleton rows | Partition into cycles | Solve each cycle

- Solve singleton rows directly
- Eliminate the singleton columns
- Partition into cycles
- Solve each cycle
- Solve singleton columns

# Convert to Cycle Matrix



Eliminate singleton rows — Partition into cycles — Solve each cycle

- Solve singleton rows directly
- Eliminate the singleton columns
- Partition into cycles
- Solve each cycle
- Solve singleton columns

# Solve Cycle Matrix

- Scale Cycle Matrix as:

$$W = \begin{bmatrix} 1 & -\alpha_1 & 0 & \ldots & 0 \\ 0 & 1 & -\alpha_2 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & 1 & -\alpha_{n-1} \\ -\alpha_n & 0 & \ldots & 0 & 1 \end{bmatrix}$$

- Solve $Wu = d$ as:

$$u_1 = \frac{\sum_{j=1}^{n}(d_j \beta_{j-1})}{1 - \beta_n} \qquad (0)$$

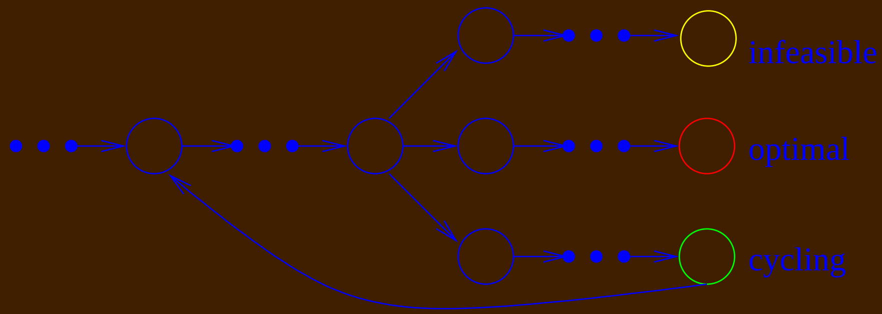
$$u_{i+1} = \frac{1 - \sum_{j=1}^{i}(d_j \beta_{j-1})}{\beta_i} \qquad (1)$$

where $\beta_k = \prod_{i=1}^{k} \alpha_i$.

- Hybrid method

# Interval Based Simplex

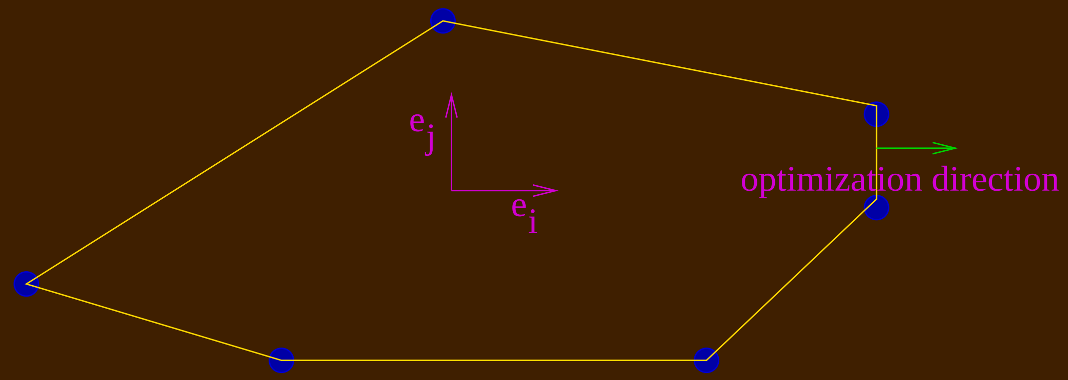
- Interval Arithmetic
  - Possible underapproximation because of rounding off.
  - Represents $x$ by $[\underline{x}, \overline{x}]$.
  - Possible because of the effi cient and accurate linear system solver.
  - Comparison of interval numbers might be unclear.
- Pivot Selection




  - Try all possible favorable pivots
  - Cycling? Hashtable
  - Infeasible? Drop
- Optimal Solution
  - A set of *possible* optimal bases.
  - A lower/upper bound of the optimal value.

# Projection Algorithm

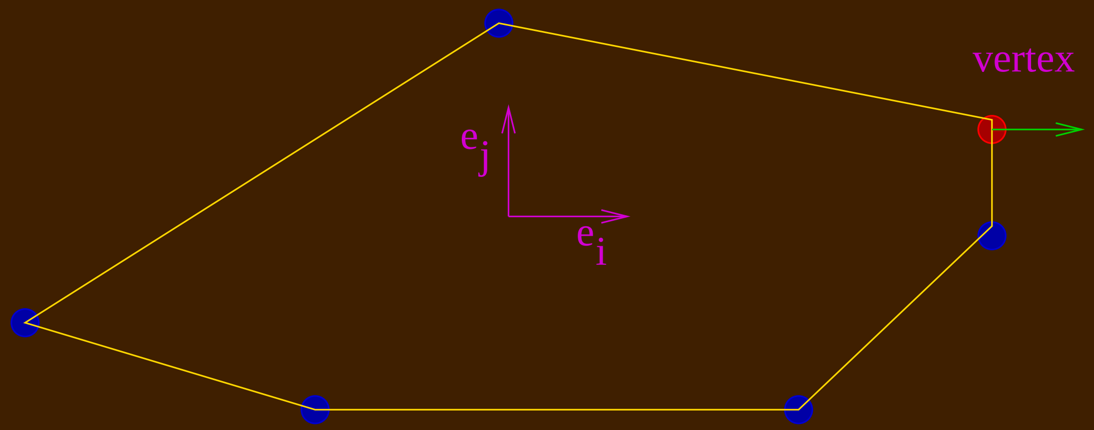
- Basic Ideas
  - Find vertex of projection polygon using linear programming.
  - Find normal of projection polygon by the feasibility of basis.




- Slightly Over Approximated

# Projection Algorithm

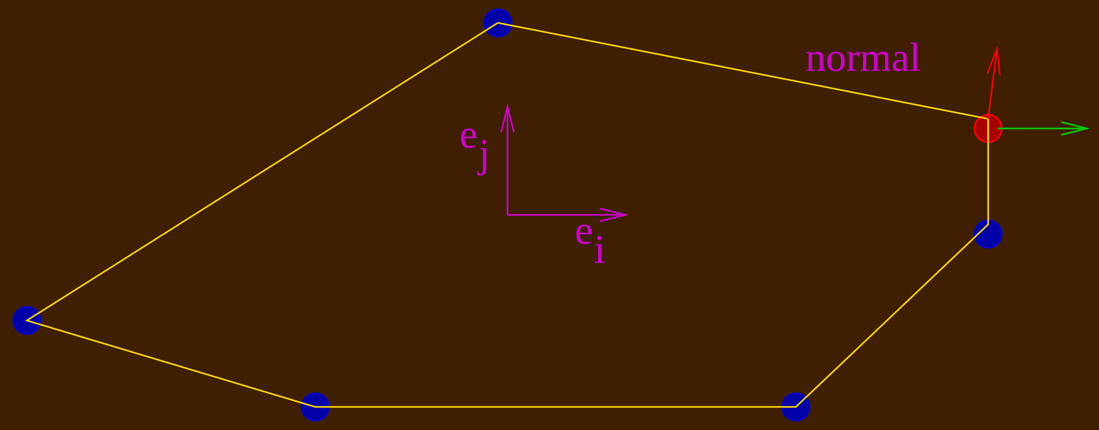
- Basic Ideas
  - Find vertex of projection polygon using linear programming.
  - Find normal of projection polygon by the feasibility of basis.




- Slightly Over Approximated

# Projection Algorithm

- Basic Ideas
  - Find vertex of projection polygon using linear programming.
  - Find normal of projection polygon by the feasibility of basis.




- Slightly Over Approximated

# Experience

## Three Dimensional Van der Pol's Oscillator

- It is modelled by ODE:

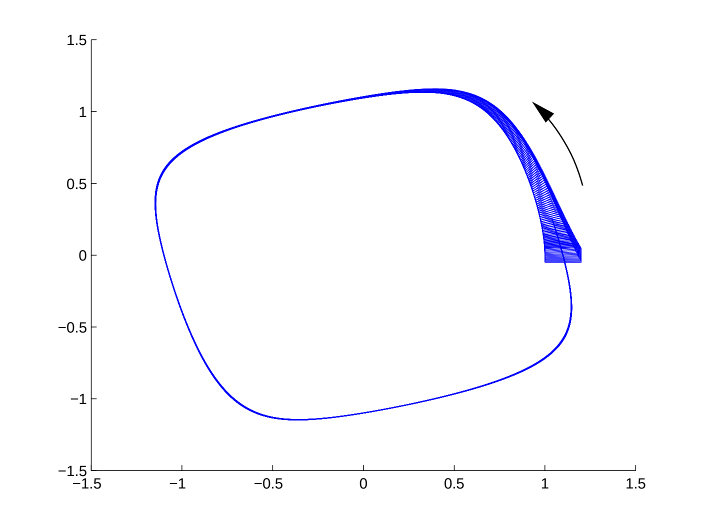
$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = \begin{pmatrix} -y - x^3 + x \\ x - y^3 + y \\ 2x^2 - 2z \end{pmatrix}$$

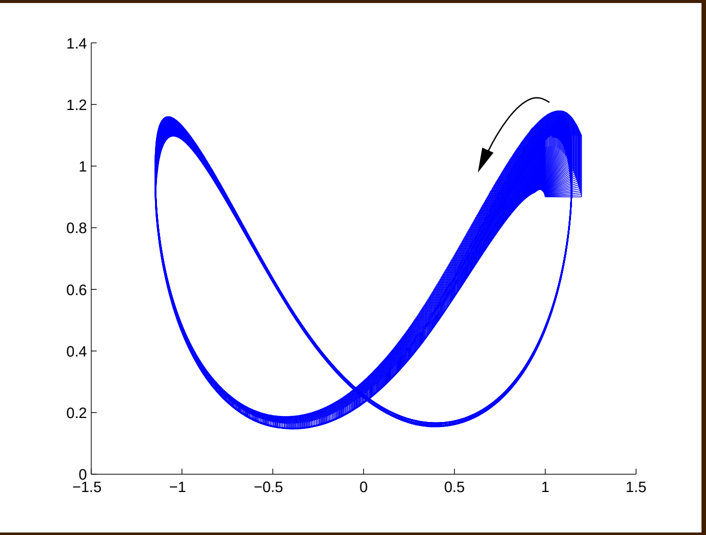
  with the initial region is the cube with the diagonal

$$[(1.0, -0.05, 0.9),\ (1.2, 0.05, 1.1)]$$

- Coho Steps

# Experience




- An invariance is established by showing the region at the end is contained in the initial region.
- Error bond of linear program solver is less than $\sqrt{ulp}$.
- Pivot selection is effi cient, most of pivots have unique branch.

# Summary

- Coho: a Reachability Tool for Circuit Verifi cation
- A Robust Linear Program Solver Using Interval Arithmetic
- A $O(n)$ Linear System Solver
- Projection Algorithm
- Future Work
  - Apply to more examples
  - Reduce error bound
  - Using arbitrary precision number
  - Linearize model automatically
  - Projectagon operations
  - Improve effi ciency