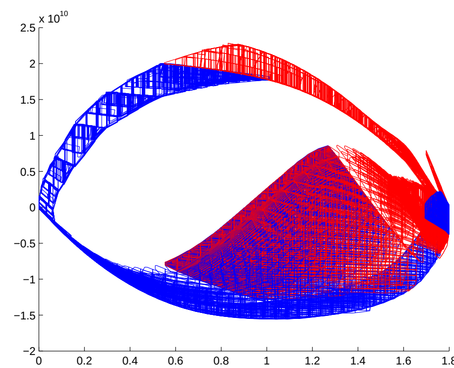
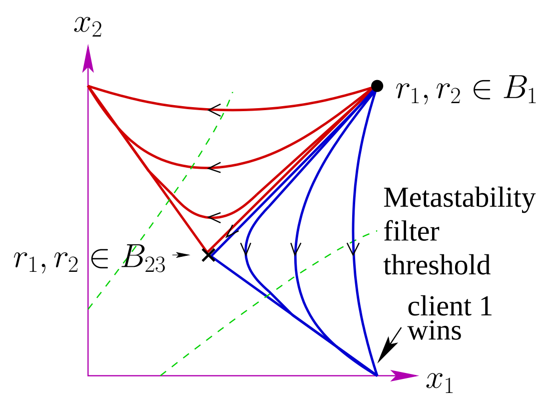
Formal Verification of an Arbiter

Chao Yan & Mark Greenstreet & Jochen Eisinger

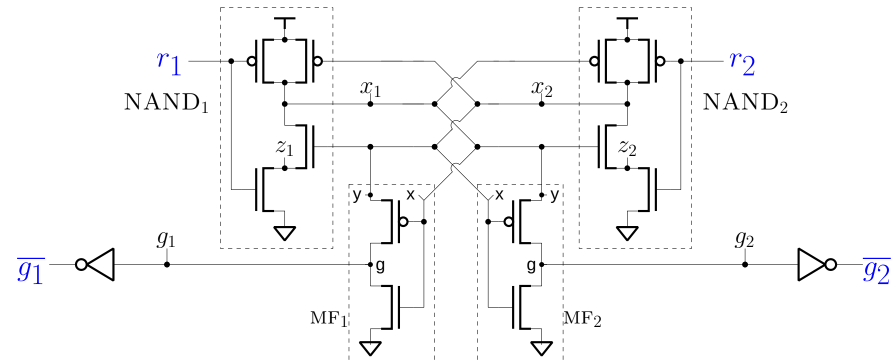
The University of British Columbia





Outline

- Circuit-Level Verification
- Arbiter
  - Specification
  - Previous work
  - An arbiter circuit
- Verification as Reachability
  - Coho
  - Stiffness
- Almost-Surely Verification
- Conclusion and Future Work

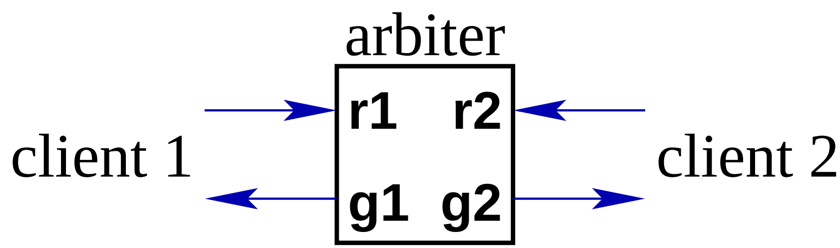


Circuit-Level Verification

- Digital circuit verification
  - Show that a circuit in an analog-model implements the desired discrete behavior.
- Why do Circuit-Level Verification?
  - Digital design has become relatively low error:
  - Circuit-level bugs remain a problem:
    - SPICE is still the main validation tool, and it doesn't scale.
    - Deep-submicron circuit effects undermine digital abstractions.
    - Hard/impossible to simulate bugs.
- Example
  - Synchronoizer τ doesn't scale with $FO4$.
  - Deep metastability beyond resolution of HSPICE.
  - How can we make sure our designs don't have such "scaling bugs"?

Arbiters

- A Black-Box View

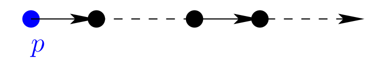


- Previous Work
  - Kurshan & McMillan (IEEE TCAD 1991) verified an nMOS arbiter.
    - Assumed inputs make instantaneous transitions.
    - This assumption greatly reduces the size of the reachable space.
  - No perfect arbiter can be built
    - Hurtado 1975, Marino 1981, Chapiro 1984, Mendler & Stroup 1993.
- Challenges
  - Formal specification
  - Realistic device and input models
  - Stiffness
  - Metastable behaviour

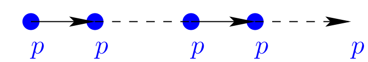
Discrete Specification

- LTL logic

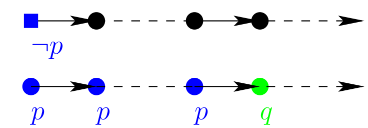
  p: p holds in the current state.

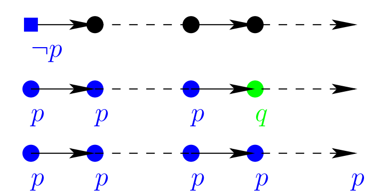
  $\Box p$: p holds this and all subsequent states.

  $p \, \widehat{\mathbf{U}} \, q$: if p holds in the current state, p will continue to hold until a state in which q holds.

  $p \, \widehat{\mathbf{W}} \, q$: if p holds in the current state, p will continue to hold forever or until a state in which q holds.

- Specification

Discrete Specification

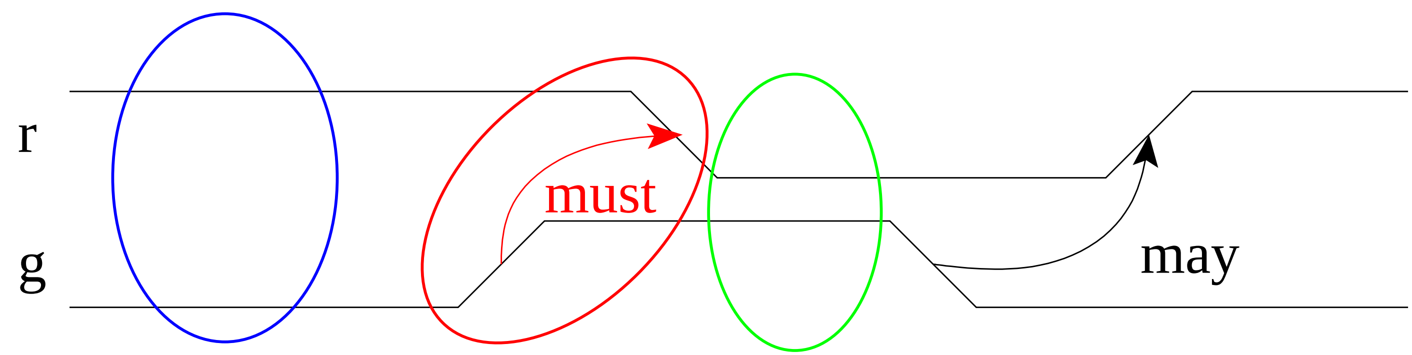
- LTL logic
- Specification

Initially:

$$\forall i \in \{1,2\}.\ \neg \mathsf{r}_i \wedge \neg \mathsf{g}_i$$

Assume (environment controls r_1 and r_2):

$$\forall i \in \{1,2\}.\quad \Box(\mathsf{r}_i\ \widehat{\mathbf{W}}\ \mathsf{g}_i) \wedge \Box(\neg \mathsf{r}_i\ \widehat{\mathbf{W}}\ \neg \mathsf{g}_i) \wedge \Box(\mathsf{g}_i\ \widehat{\mathbf{U}}\ \neg \mathsf{r}_i)$$



r holds until g $\uparrow$ $\quad g \uparrow \rightarrow r \downarrow$ $\quad$ r holds until g $\downarrow$

Discrete Specification

- LTL logic
- Specification

Initially:

$$\forall i \in \{1,2\}.\ \neg \mathsf{r}_i \wedge \neg \mathsf{g}_i$$

Assume (environment controls r_1 and r_2):

$$\forall i \in \{1,2\}. \quad \Box(\mathsf{r}_i\ \widehat{\mathbf{W}}\ \mathsf{g}_i)\ \wedge\ \Box(\neg \mathsf{r}_i\ \widehat{\mathbf{W}}\ \neg \mathsf{g}_i)$$
$$\wedge \quad \Box(\mathsf{g}_i\ \widehat{\mathbf{U}}\ \neg \mathsf{r}_i)$$

Guarantee (arbiter controls g_1 and g_2):

Handshake:

$$\forall i \in \{1,2\}.\ \Box(\neg \mathsf{g}_i\ \widehat{\mathbf{W}}\ \mathsf{r}_i)\ \wedge\ \Box(\mathsf{g}_i\ \widehat{\mathbf{W}}\ \neg \mathsf{r}_i)$$

Mutual Exclusion:

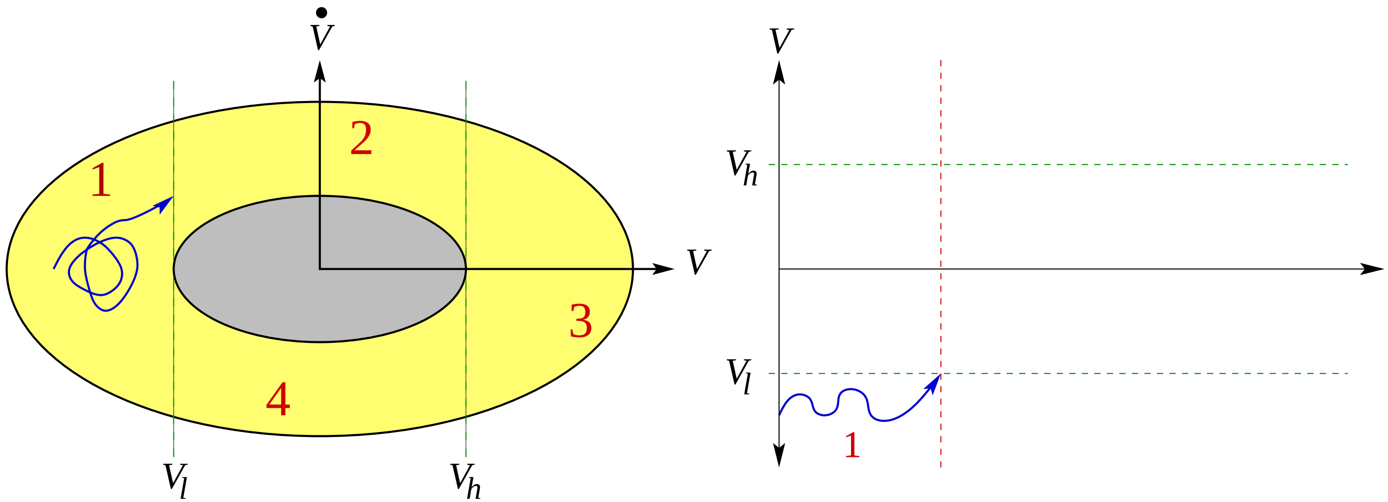
$$\Box \neg(\mathsf{g}_1 \wedge \mathsf{g}_2)$$

Liveness:

$$\forall i \in \{1,2\}.\ (\Box(\mathsf{r}_i\ \widehat{\mathbf{U}}\ \mathsf{g}_i)) \wedge (\Box(\neg \mathsf{r}_i\ \widehat{\mathbf{U}}\ \neg \mathsf{g}_i))$$

Continuous-Time Logic

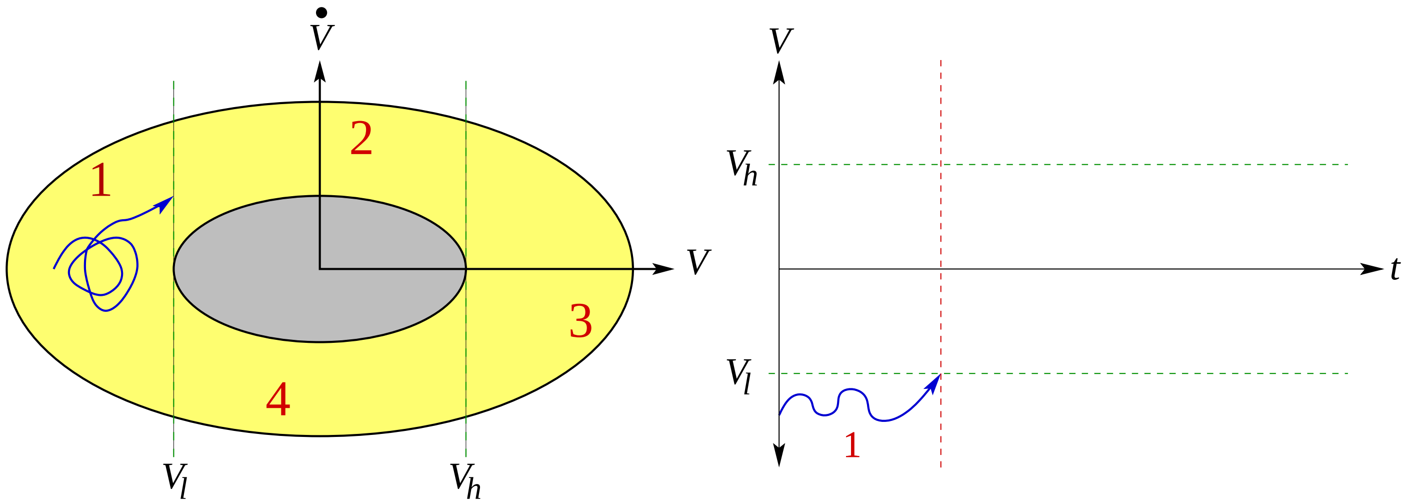
- Brockett's Annulus



- Map continuous trajectories to discrete sequences.
  - Region 1 represents a logical low signal.
  - Region 2 represents a monotonically rising signal.
  - Region 3 represents a logical high signal.
  - Region 4 represents a monotonically falling signal.
  - Brockett's annulus allows entire families of signals to be specified.

Continuous-Time Logic

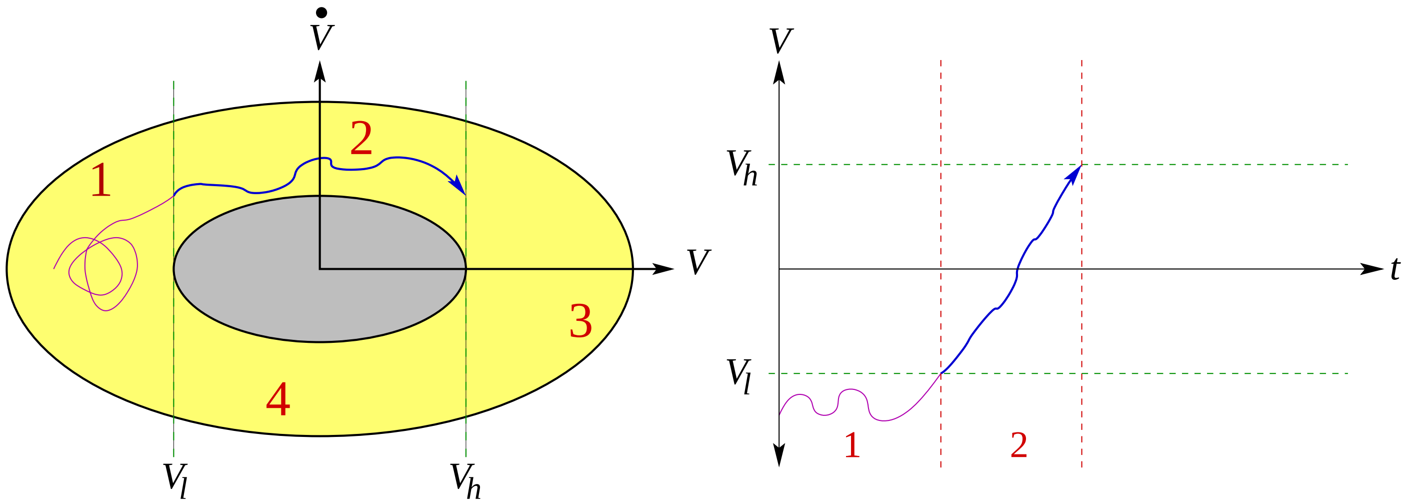
- Brockett's Annulus



  - Map continuous trajectories to discrete sequences.
  - Region 1 represents a logical low signal.
  - Region 2 represents a monotonically rising signal.
  - Region 3 represents a logical high signal.
  - Region 4 represents a monotonically falling signal.
  - Brockett's annulus allows entire families of signals to be specified.

Continuous-Time Logic

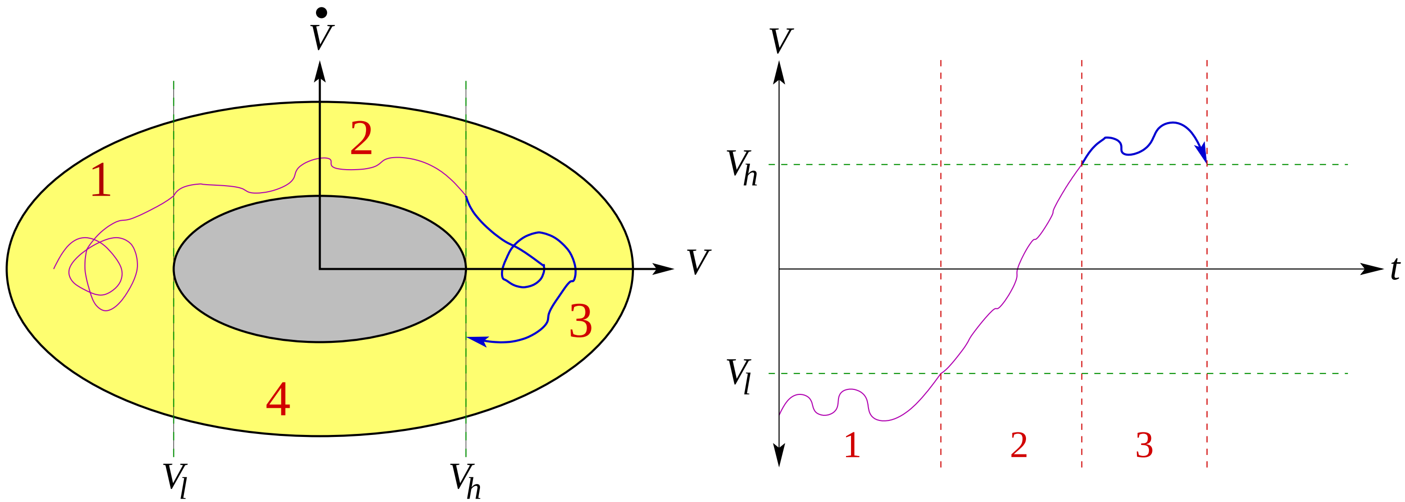
- Brockett's Annulus



  - Map continuous trajectories to discrete sequences.
  - Region 1 represents a logical low signal.
  - Region 2 represents a monotonically rising signal.
  - Region 3 represents a logical high signal.
  - Region 4 represents a monotonically falling signal.
  - Brockett's annulus allows entire families of signals to be specified.

Continuous-Time Logic

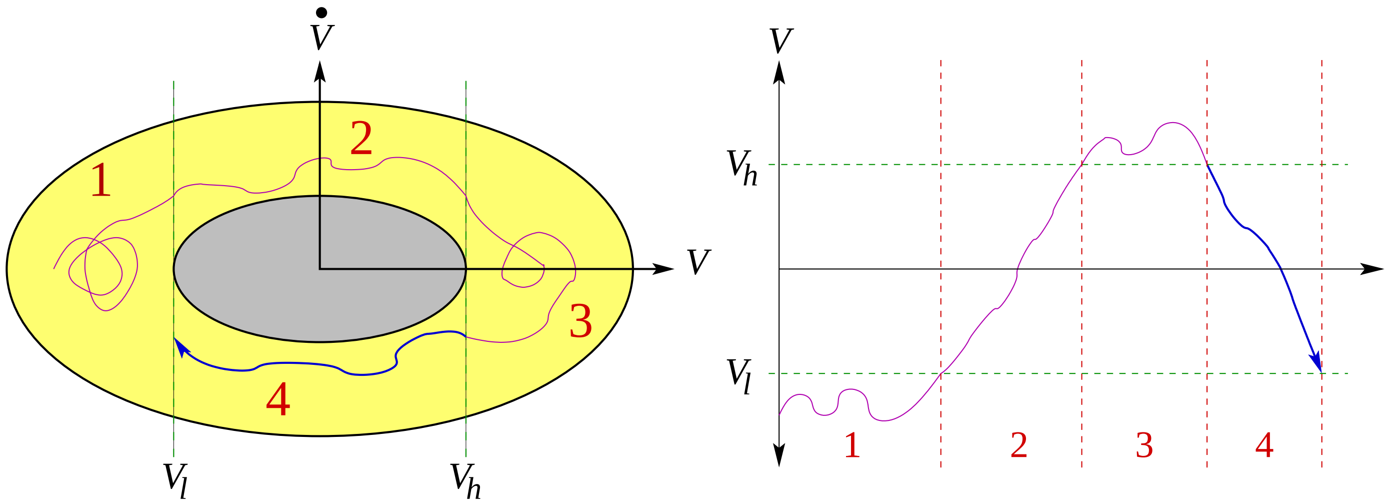
- Brockett's Annulus



  - Map continuous trajectories to discrete sequences.
  - Region 1 represents a logical low signal.
  - Region 2 represents a monotonically rising signal.
  - Region 3 represents a logical high signal.
  - Region 4 represents a monotonically falling signal.
  - Brockett's annulus allows entire families of signals to be specified.

Continuous-Time Logic

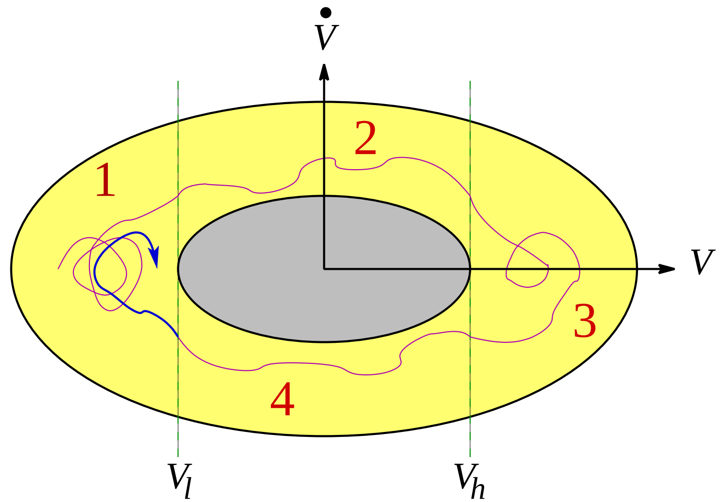
- Brockett's Annulus

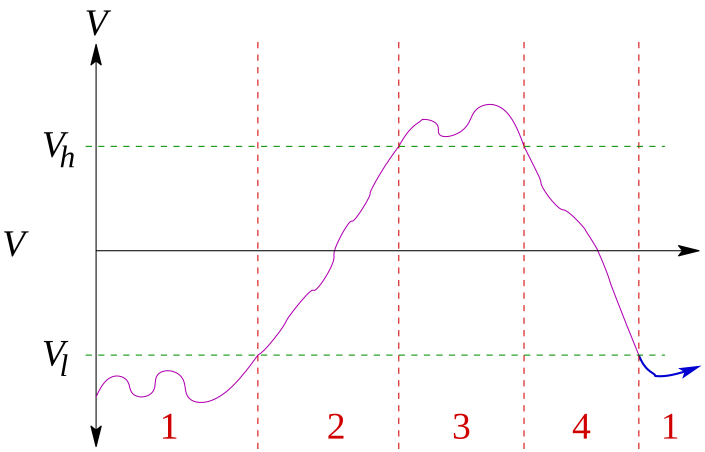


  - Map continuous trajectories to discrete sequences.
  - Region 1 represents a logical low signal.
  - Region 2 represents a monotonically rising signal.
  - Region 3 represents a logical high signal.
  - Region 4 represents a monotonically falling signal.
  - Brockett's annulus allows entire families of signals to be specified.

Continuous-Time Logic

- Brockett's Annulus





  - Map continuous trajectories to discrete sequences.
  - Region 1 represents a logical low signal.
  - Region 2 represents a monotonically rising signal.
  - Region 3 represents a logical high signal.
  - Region 4 represents a monotonically falling signal.
  - Brockett's annulus allows entire families of signals to be specified.

Continuous-Time Logic

- Brockett's Annulus

Initially:

$\forall i \in \{1, 2\}.\ \neg r_i \wedge \neg g_i$

$\forall i \in \{1, 2\}.\ B_1(r_i) \wedge B_1(g_i)$

Continuous-Time Logic

- Brockett's Annulus

Initially:

$$\forall i \in \{1,2\}.\ B_1(r_i) \wedge B_1(g_i)$$

Assume (environment controls r_1 and r_2):

$$\begin{array}{lll} \forall i \in \{1,2\}. & & \Box(r_i \,\widehat{\mathbf{W}}\, g_i) \;\wedge\; \Box(\neg r_i \,\widehat{\mathbf{W}}\, \neg g_i) \\ & \wedge & \Box(g_i \,\widehat{\mathbf{U}}\, \neg r_i) \\ \forall i \in \{1,2\}. & & \Box(B_3(r_i) \,\widehat{\mathbf{W}}\, B_{2,3}(g_i)) \;\wedge\; \Box(B_1(r_i) \,\widehat{\mathbf{W}}\, B_{4,1}(g_i)) \\ & \wedge & \Box(B_3(g_i) \,\widehat{\mathbf{U}}\, B_4(r_i)) \end{array}$$

Continuous-Time Logic

- Brockett's Annulus

Initially:

$$\forall i \in \{1,2\}.\ B_1(r_i) \wedge B_1(g_i)$$

Assume (environment controls r_1 and r_2):

$$\forall i \in \{1,2\}.\quad \Box(B_3(r_i)\ \widehat{\mathbf{W}}\ B_{2,3}(g_i)) \ \wedge\ \Box(B_1(r_i)\ \widehat{\mathbf{W}}\ B_{4,1}(g_i))$$
$$\wedge\quad \Box(B_3(g_i)\ \widehat{\mathbf{U}}\ B_4(r_i))$$

Guarantee (arbiter controls g_1 and g_2):

Handshake:

$$\forall i \in \{1,2\}.\ \Box(B_1(g_i)\ \widehat{\mathbf{W}}\ B_{2,3}(r_i)) \ \wedge\ \Box(B_3(g_i)\ \widehat{\mathbf{W}}\ B_{4,1}(r_i))$$

Mutual Exclusion:

$$\Box\neg(B_{2,3}(g_1) \wedge B_{2,3}(g_2))$$

Liveness:

$$\forall i \in \{1,2\}.\ (\Box(B_3(r_i)\ \widehat{\mathbf{U}}\ B_{2,3}(g_i))) \wedge (\Box(B_1(r_i)\ \widehat{\mathbf{U}}\ B_{4,1}(g_i)))$$

Continuous-Time Logic

- Brockett's Annulus
- Specify Metastable Behaviours
  - *Almost-surely* version of LTL "always" operator
    - $\phi \models \Box_Z S \equiv (\phi \models (\Box S) \vee ((\phi \in Z) \ \wedge \ (\mu(Z) = 0))$
    - A trajectory ϕ satisfies $\Box_Z S$ iff S holds everywhere along ϕ, or if ϕ is in a negligible set, Z.
    - The probability of S holding everywhere along ϕ is equal to 1.
  - $\alpha - \mathrm{insensitivity}$
    - arbiter's clients do not act as feedback controllers
    - α-ins $\Rightarrow \ (\Box_Z (B_3(r_i) \ \widehat{\mathbf{U}} \ B_{2,3}(g_i)))$

Continuous Specification

Initially:

$$\forall i \in \{1,2\}.\ B_1(r_i) \wedge B_1(g_i)$$

Assume (environment controls r_1 and r_2):

$$\begin{aligned} \forall i \in \{1,2\}. \quad & \Box(B_3(r_i)\ \widehat{\mathbf{W}}\ B_{2,3}(g_i)) \ \wedge\ \Box(B_1(r_i)\ \widehat{\mathbf{W}}\ B_{4,1}(g_i)) \\ \wedge \quad & \Box(B_3(g_i)\ \widehat{\mathbf{U}}\ B_4(r_i)) \end{aligned}$$

Guarantee (arbiter controls g_1 and g_2):

Handshake:

$$\forall i \in \{1,2\}.\ \Box(B_1(g_i)\ \widehat{\mathbf{W}}\ B_{2,3}(r_i)) \ \wedge\ \Box(B_3(g_i)\ \widehat{\mathbf{W}}\ B_{4,1}(r_i))$$

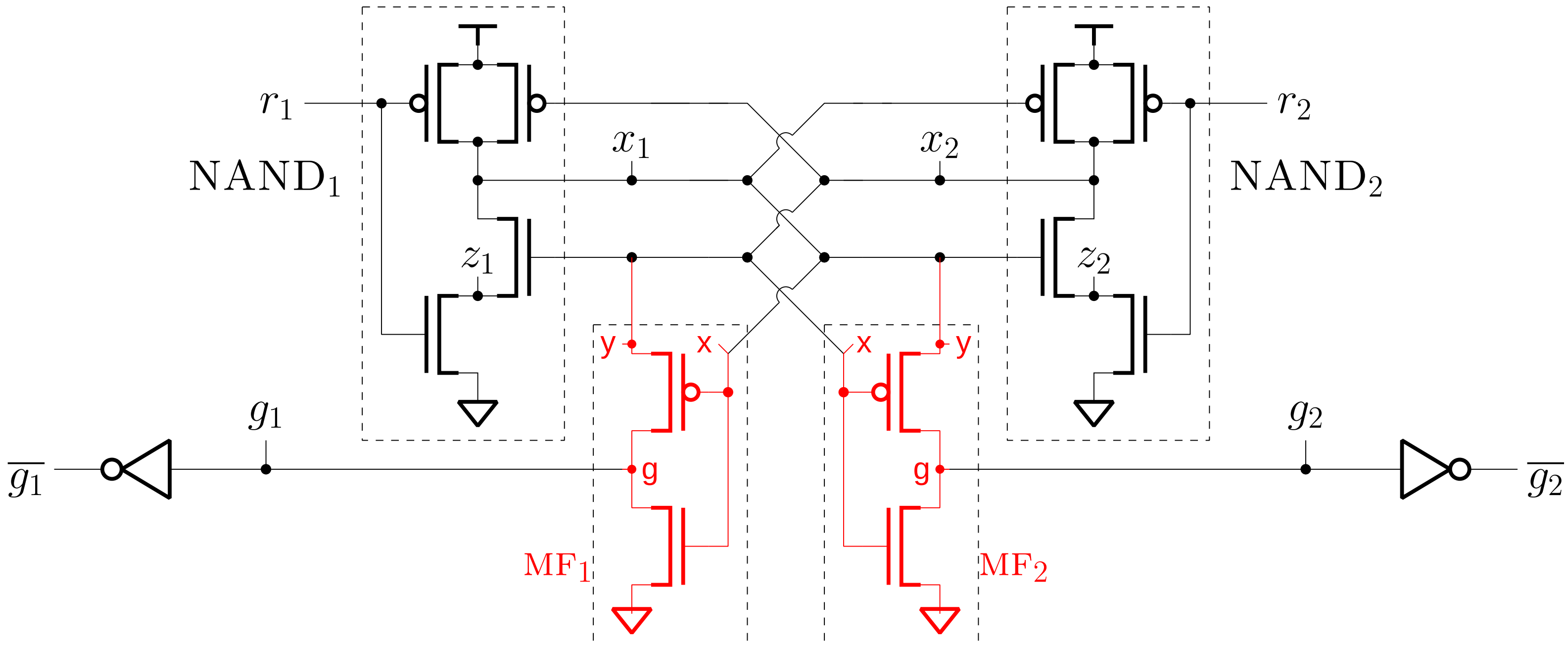
Mutual Exclusion:

$$\Box\neg(B_{2,3}(g_1) \wedge B_{2,3}(g_2))$$

Liveness:

$$\begin{aligned} & \forall i \in \{1,2\}. \\ & \qquad \alpha\text{-ins} \ \Rightarrow\ (\Box_Z(B_3(r_i)\ \widehat{\mathbf{U}}\ B_{2,3}(g_i))) \\ & \wedge \quad (B_3(r_i)\ \widehat{\mathbf{U}}\ (B_{2,3}(g_i) \vee B_3(r_{\sim i}))) \\ & \wedge \quad (\Box(B_1(r_i)\ \widehat{\mathbf{U}}\ B_{4,1}(g_i))) \end{aligned}$$

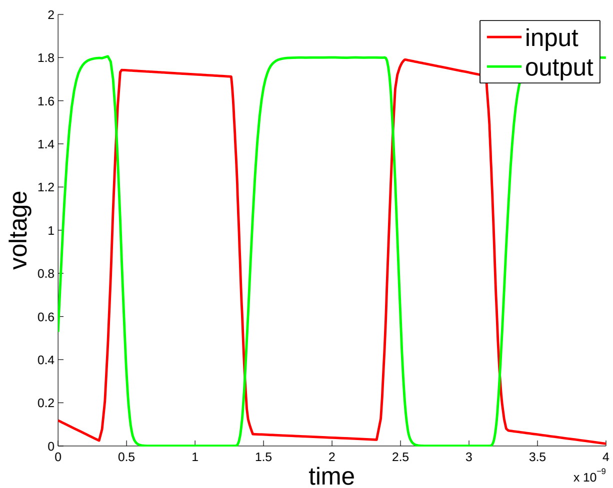
Arbiter Circuit

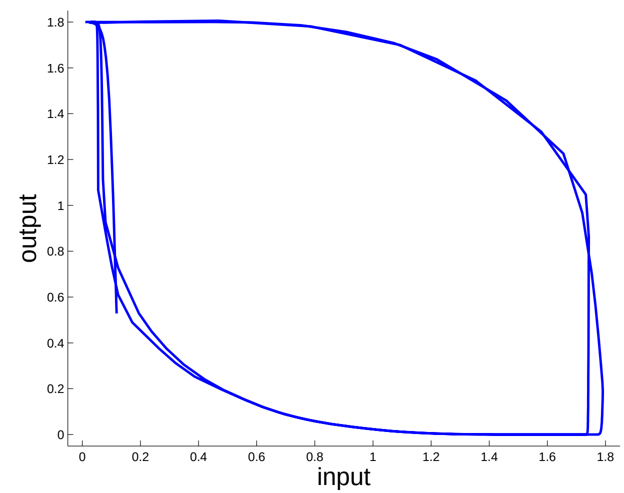


- Based on a SR-latch
- The *metastability filters* ensure that no grant is asserted until metastability has resolved.

Verification as Reachability

- Phase-space view of circuit behaviour
  - Waveforms $\longrightarrow$ phase-space (reachable regions)
  - An inverter example

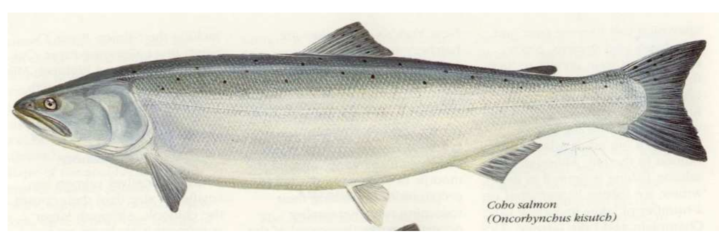




- Formal verifiation by reachability analysis
  - Specify continuous signals by Brockett annuli
  - Compute the reachable region that contains all reachable trajectories
  - Verify each signal satisfies its Brockett annulus
  - Verify LTL properties are satisfied in the reachable region

Coho: Reachability Computation Tool

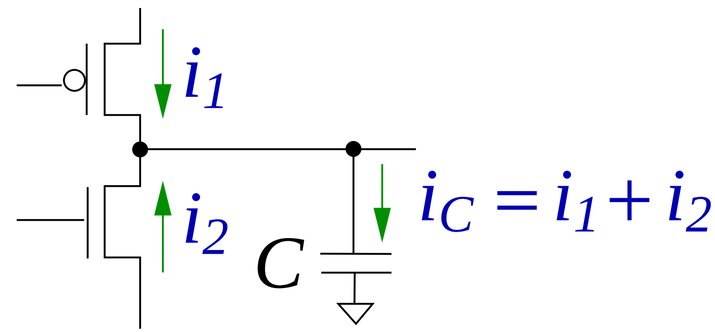
- Solving dynamic systems: linear differential inclusions.
  - Inclusions computed for neighborhood of a region.
  - Each inclusion is of the form: $\dot{v} = Av + b \pm u$, where u is an error term.
- Representing and manipulating high dimensional space: projectagon
  - Provides a tractable representation.
  - Exploits extensive algorithms for 2D computational geometry.
- All approximations overapproximate the reachable space:
  - Coho is sound for verifying safety properties.
  - False negatives are possible.



`http://pond.dnr.cornell.edu/nyfish/Salmonidae/coho_salmon.jpg`

Circuit Model

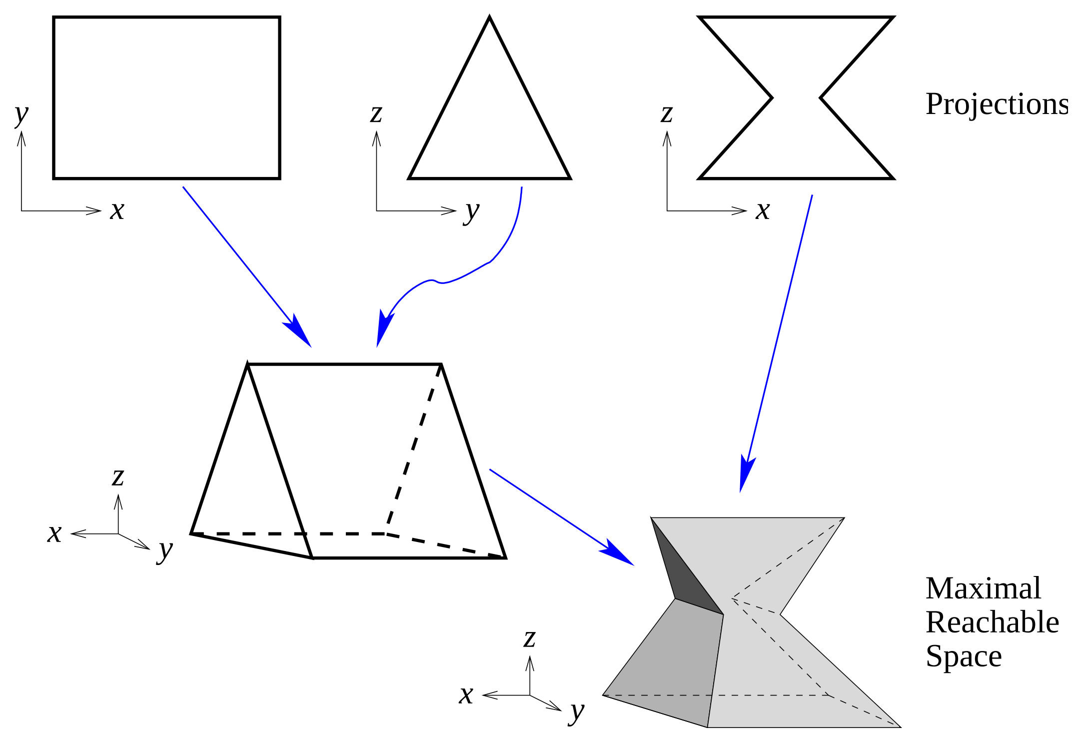
- Transistors modeled as voltage controlled current sources.
- The I_{ds} function is obtained by tabulated data from HSPICE simulations.



- At each time step, and for each projection polygon edge, Coho:
  - computes a bounding box for node voltages of each transistor.
  - computes a model of the form $i_1 = A_1 v + b_1 \pm u_1$ where u_1 is an error bound. Likewise for i_2.
  - bounds $i_c = (A_2 + A_2)v + (b_1 + b_2) \pm (u_1 + u_2)$. This produces a worst-case error bound.
- Approximate the ODEs by *linear differential inclusions*:

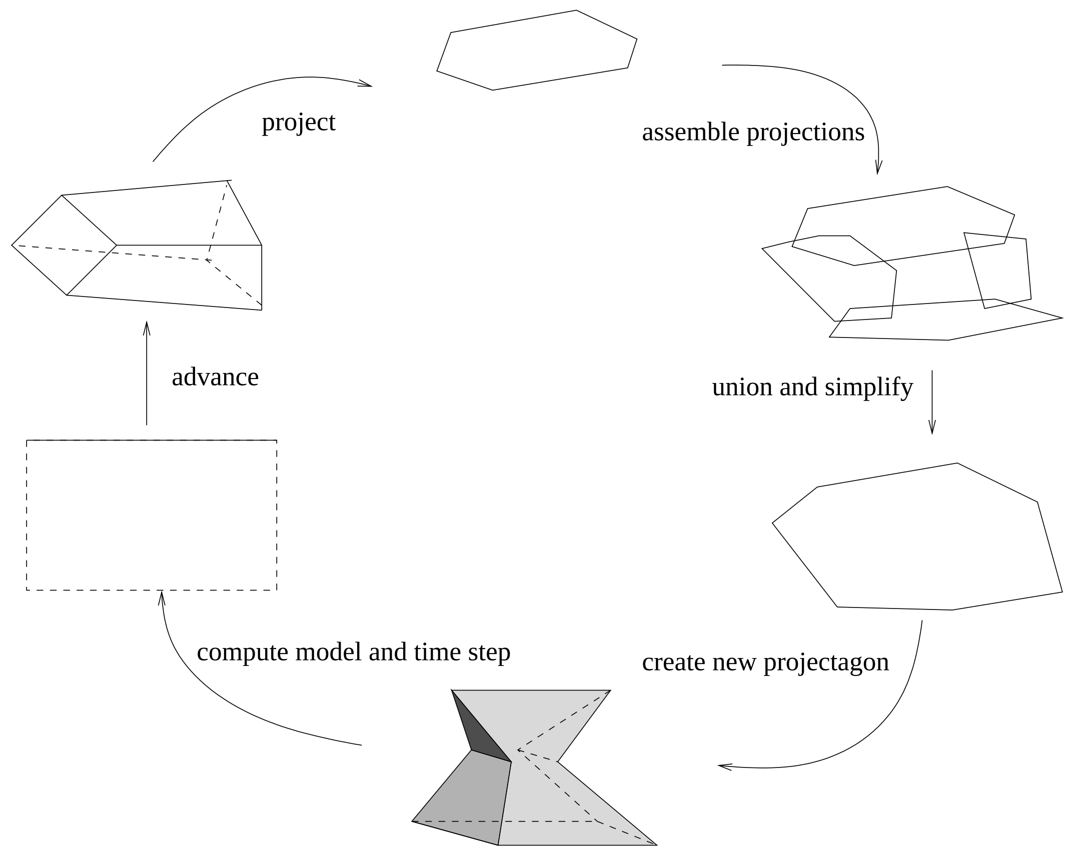
$$\boldsymbol{A}\begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{in} \end{bmatrix} + \boldsymbol{b} - \boldsymbol{u} \;\leq\; \dot{\boldsymbol{v}} \;\leq\; \boldsymbol{A}\begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{in} \end{bmatrix} + \boldsymbol{b} + \boldsymbol{u}$$

Projectagons



- Coho projects high dimensional polyhedron onto two-dimensional subspaces.
- Projectagons are efficiently manipulated using two-dimensional geometry computation algorithms.
- Projectagon faces correspond to projection polygon edges.

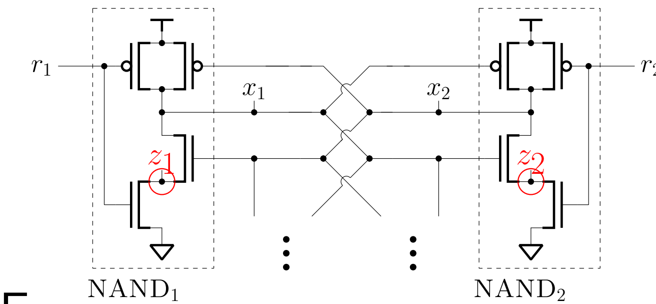
Projectagons



- A bounding projectagon is obtained by moving each face forward in time.
- The advanced face is projected onto two-dimensional subspaces to maintain the structure of projectagon.

Stiffness

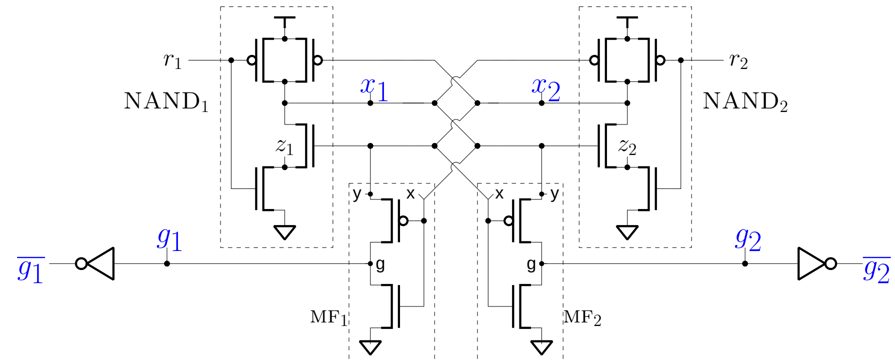
- Stiffness
  - Wide range of time constants for nodes in the arbiter
    - R $\in$ [0.6 $k\Omega$ - 13 $P\Omega$]
    - C $\in$ [0.01 pf - 0.13 pf]

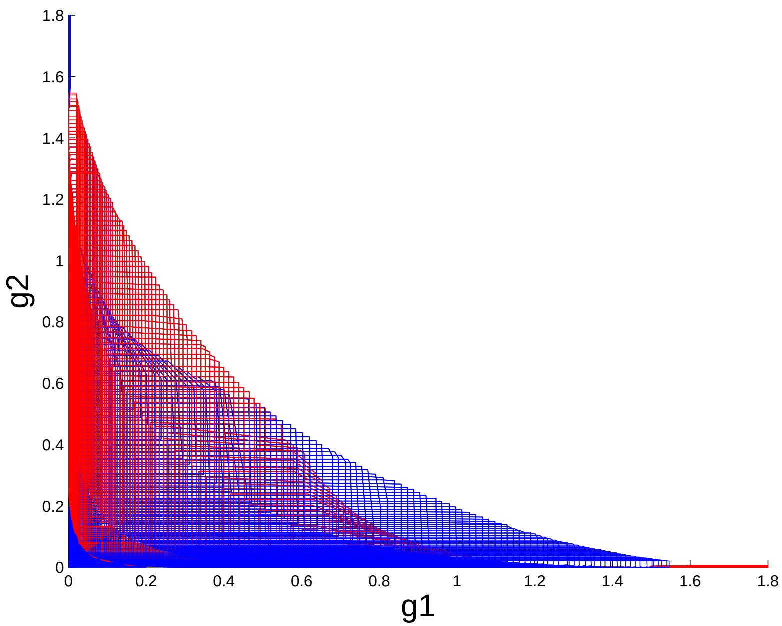


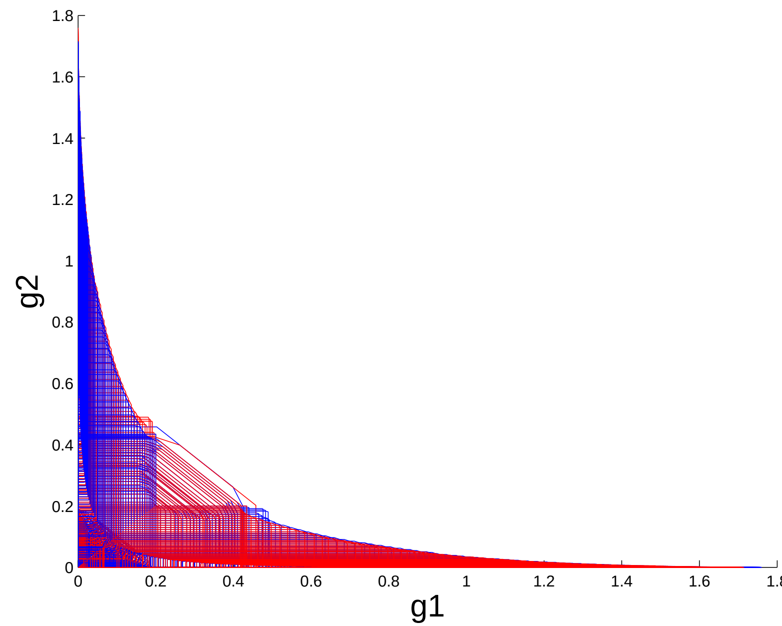
  - Ill-conditioned Jacobian matrix for ODE
    $\dot{x} = f(x)$
- Why stiffness is a problem for Coho
  - No good time-step
  - Errors: circuit model error, projectagon approximation error
- Solution
  - Changing variables
    - z converges to its equilibrium voltage $q(r, x1, x2)$ rapidly
    - Replace z with $u = z - q(r, x1, x2)$
  - Additional invariants

Result

- ➔ ● Mutual Exclusion
- ○ Brockett Annuli



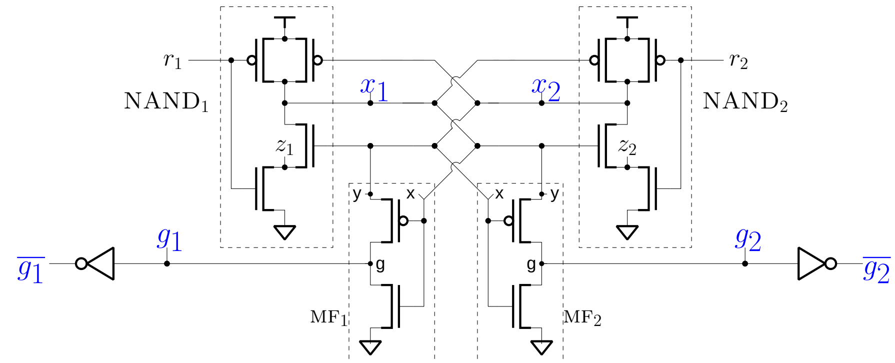


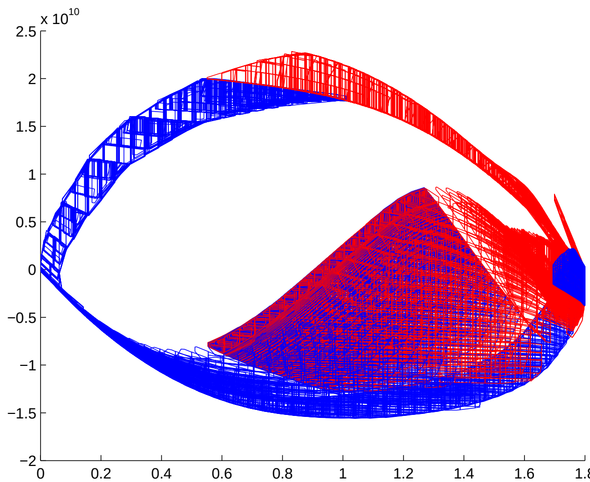


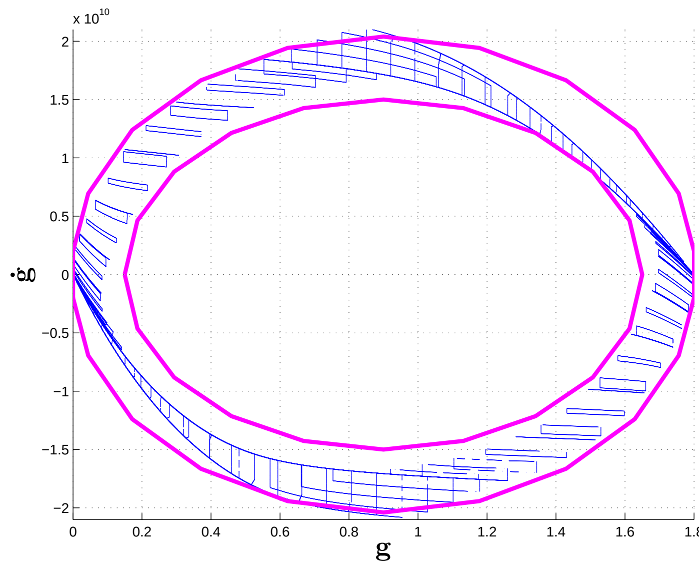
a. g_1 vs. g_2 (including B_4)   b. g_1 vs. g_2 (excluding B_4).

Result

- Mutual Exclusion
- Brockett Annuli
  - Metastability filters work as Brockett's Annulus transformers





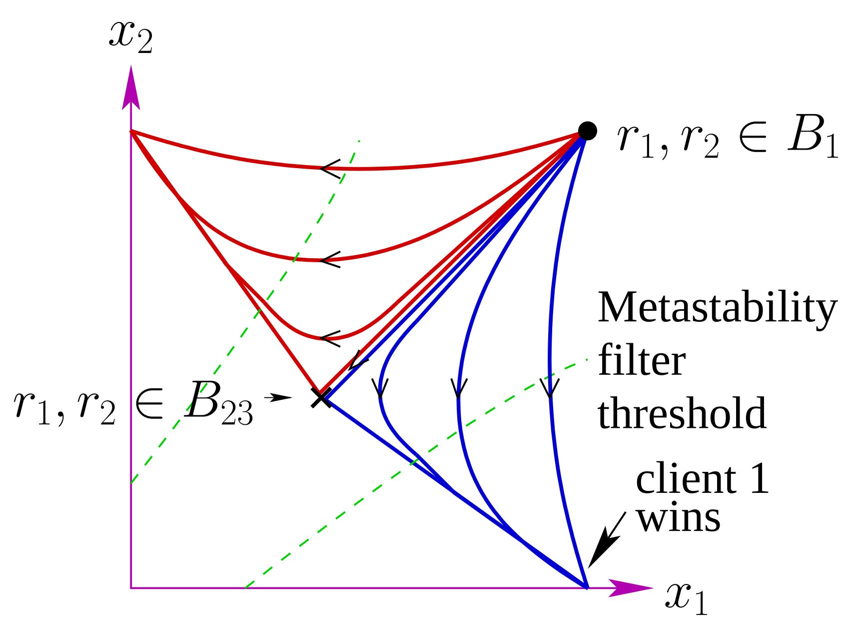


a. $\dot{x}_1$ vs. x_1

b. $\dot{\overline{g}}_1$ vs $\overline{g}_1$

Liveness

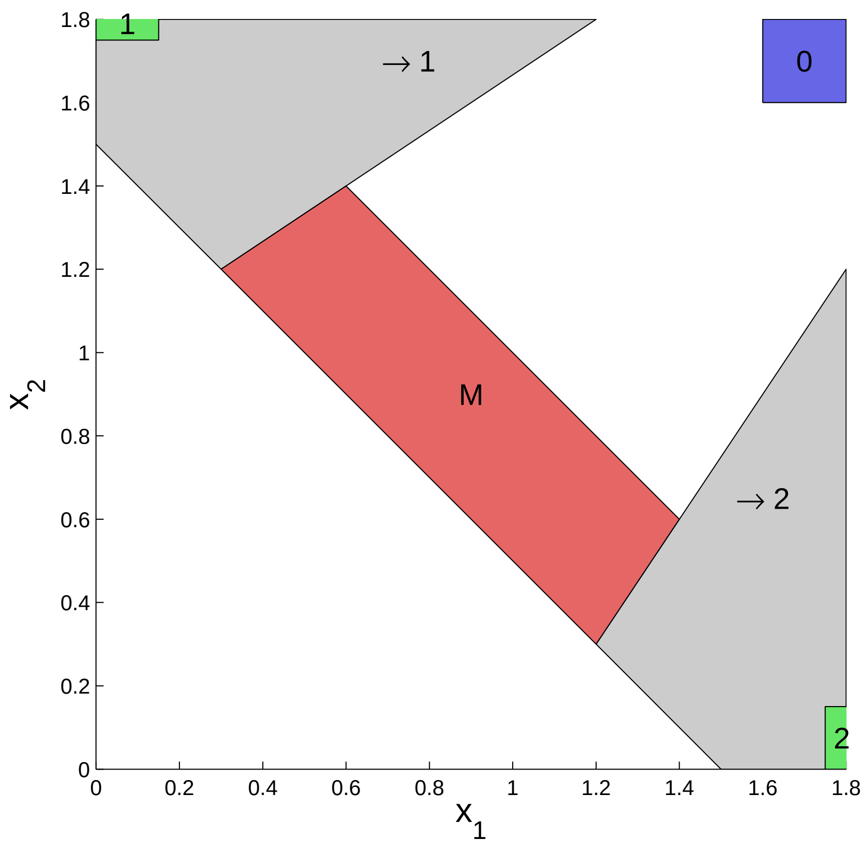
- Metastable behaviours
  - Both requests are asserted concurrently
  - Fail to show a client is eventually granted



- Bound the metastable region by Coho

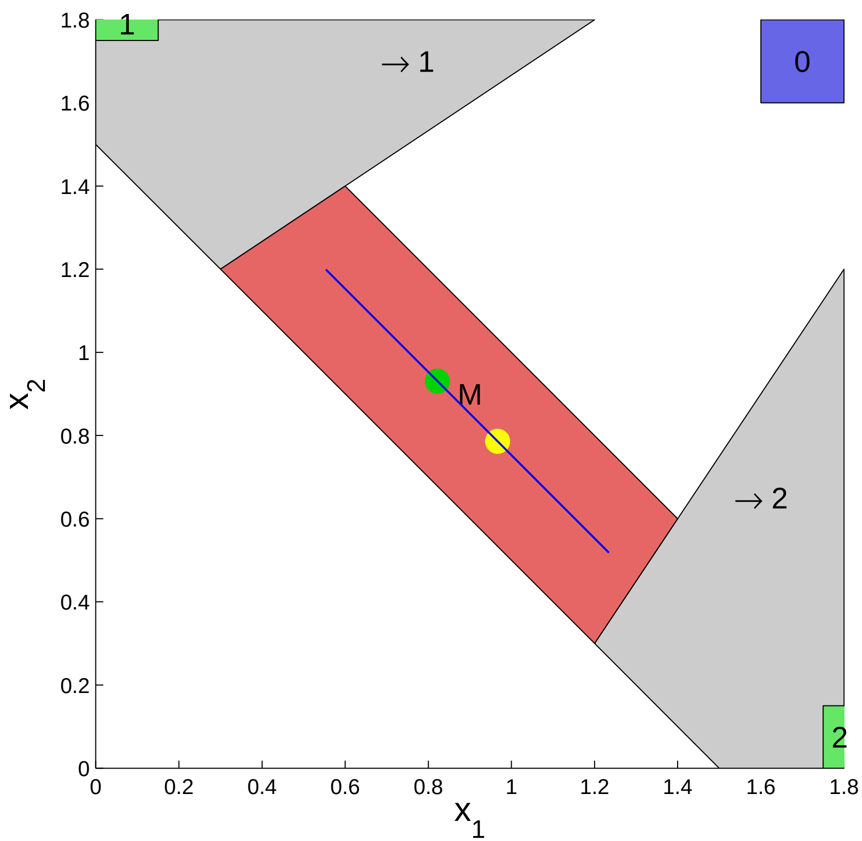
Liveness

- Metastable behaviours
- Bound the metastable region by Coho
  - Stay in region M forever because of over-approximation.
  - How to show region M is exited with probability one?



Almost-surely Approach

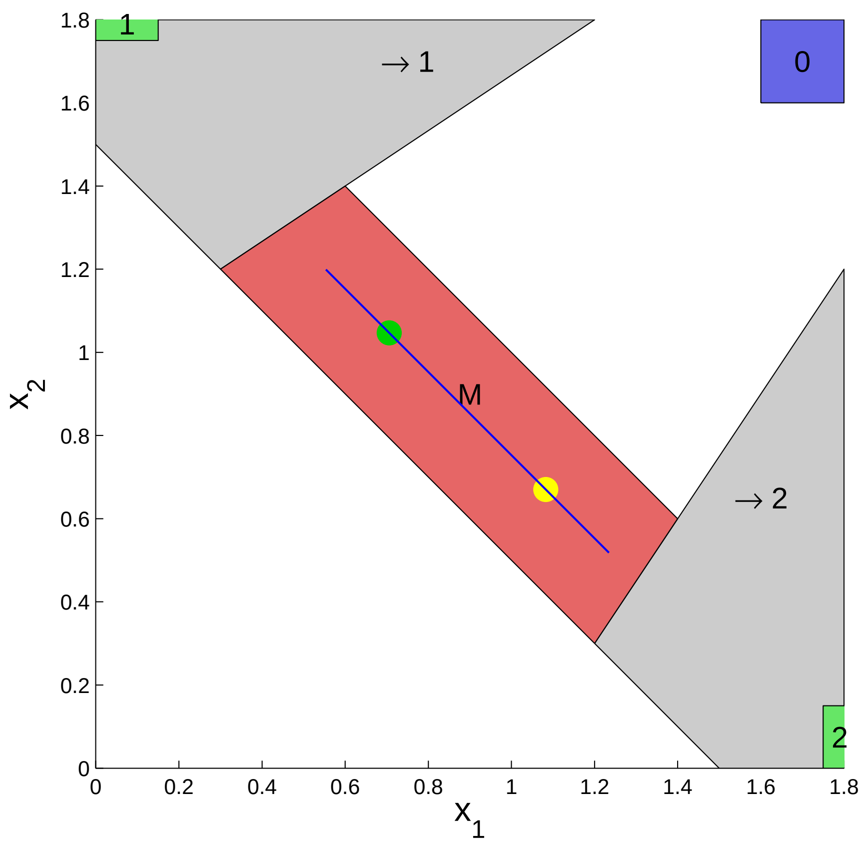
- Intuition
  - The distance of any two points on a line increases
  - There is at most one point stay in the metastable region
  - The set of trapped trajectories is of maximum dimension $d-1$



- Double Cone

Almost-surely Approach

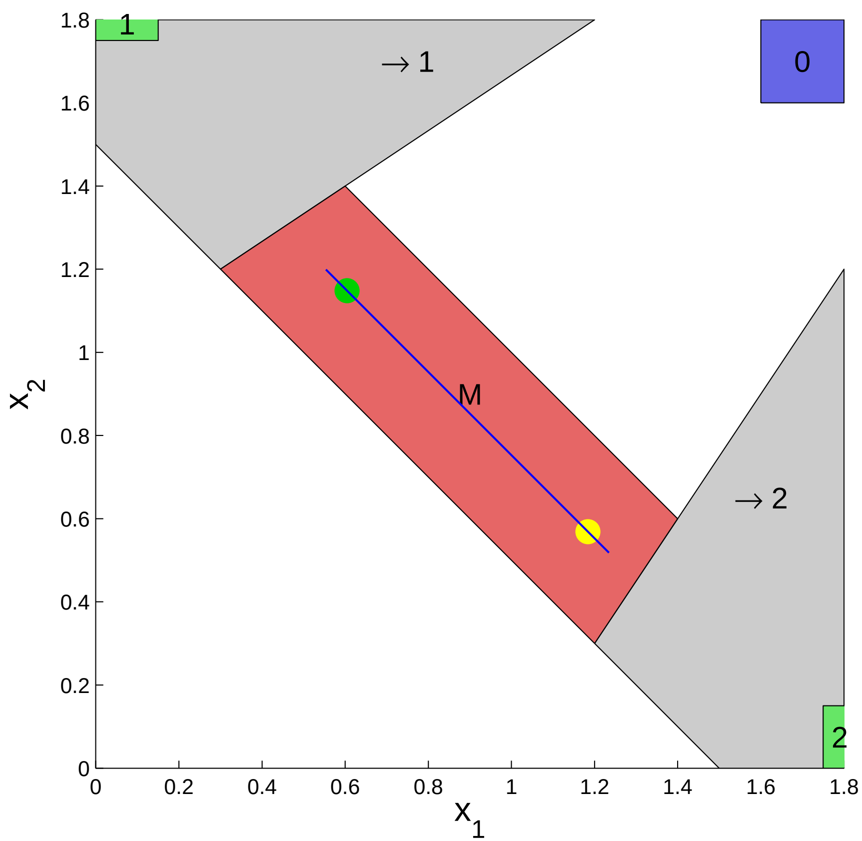
- Intuition
  - The distance of any two points on a line increases
  - There is at most one point stay in the metastable region
  - The set of trapped trajectories is of maximum dimension $d - 1$



- Double Cone

Almost-surely Approach

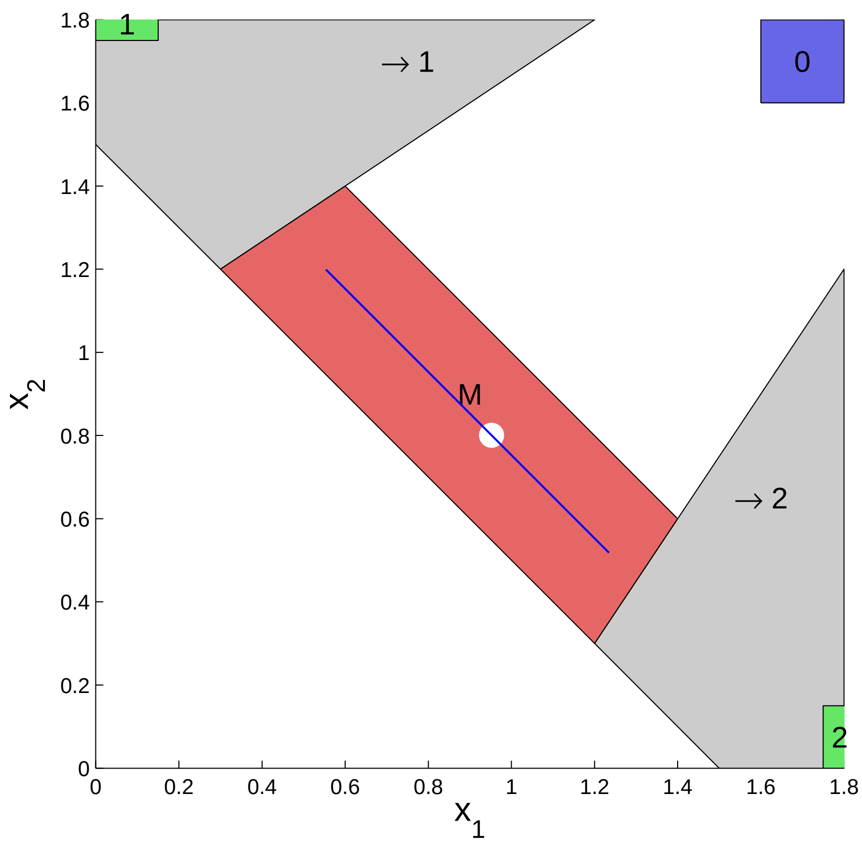
- Intuition
  - The distance of any two points on a line increases
  - There is at most one point stay in the metastable region
  - The set of trapped trajectories is of maximum dimension $d - 1$



- Double Cone

Almost-surely Approach

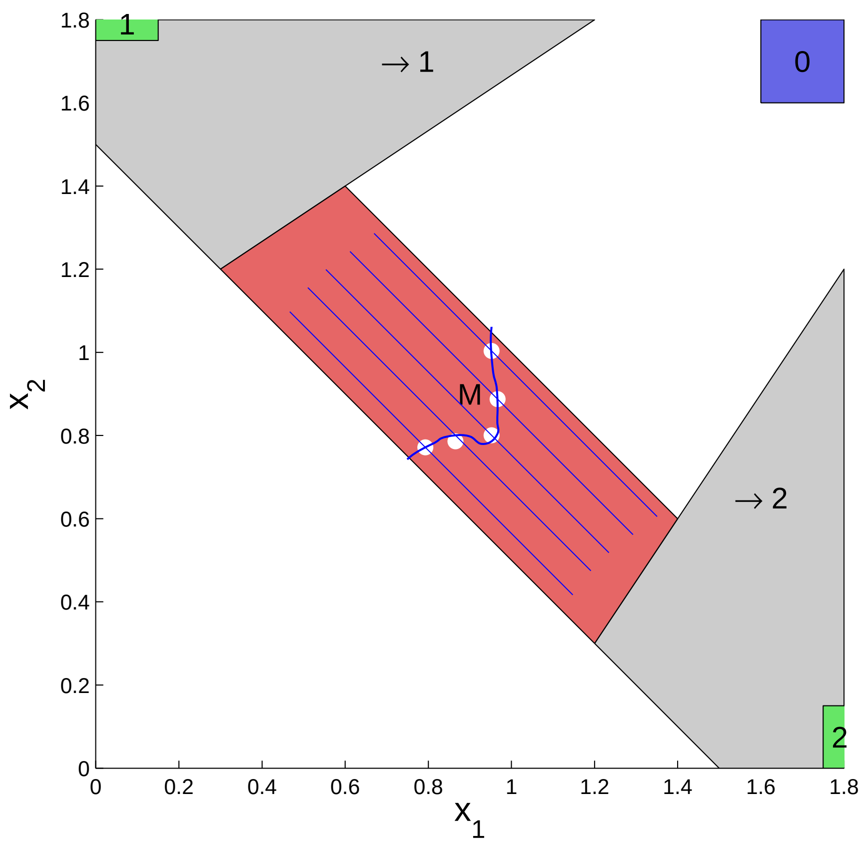
- Intuition
  - The distance of any two points on a line increases
  - There is at most one point stay in the metastable region
  - The set of trapped trajectories is of maximum dimension $d - 1$



- Double Cone

Almost-surely Approach

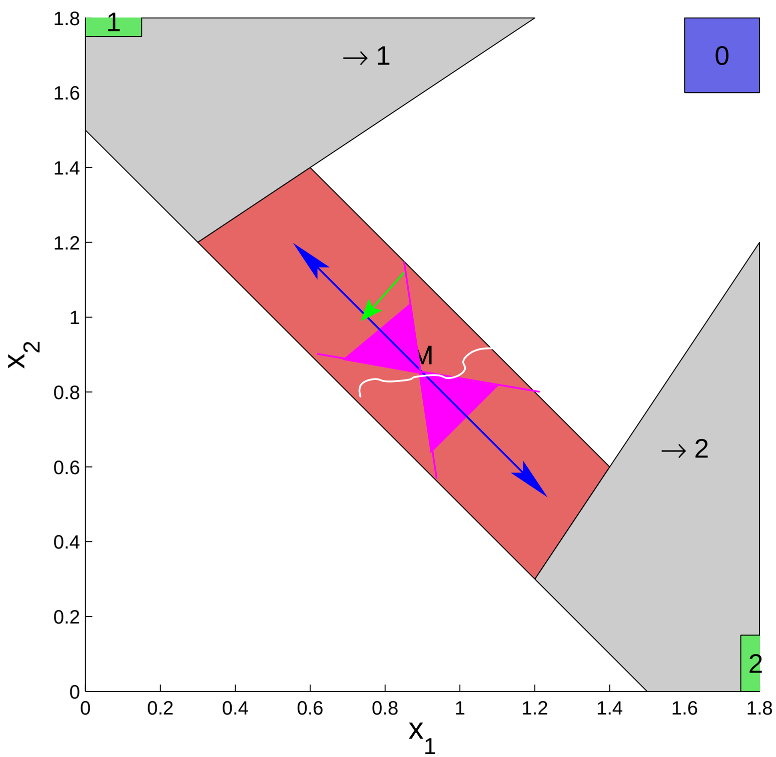
- Intuition
  - The distance of any two points on a line increases
  - There is at most one point stay in the metastable region
  - The set of trapped trajectories is of maximum dimension $d - 1$



- Double Cone

Almost-surely Approach

- Intuition
- Double Cone
    - Construct a double cone with vertex on a trajectory
    - Trajectories on boundaries flow inward
    - Trajectories in the cone diverge on the axis direction



- Sufficient condition

Almost-surely Approach

- Intuition
- Double Cone
- Sufficient condition
  - diverge property relate to the Jacobian matrix
  - details formulated in paper
- Almost Surely Verification

Almost-surely Approach

- Intuition
- Double Cone
- Sufficient condition
- Almost Surely Verification
  - use Coho to find metastable region
  - compute Jacobian matrix
  - use interval arithmetic to show divergence holds everywhere in metastable region

Conclusion and Future Work

- Conclusion
  - Specification
  - Solutions to stiffness problem
  - Almost-surely verification of metastable region
- Future Work
  - General solution for stiff system and metastable behaviours
    - An oscillator start-up verification problem
  - Formal specification for more circuits
  - Combine static (symbolic) techniques with reachability computation

Conclusion and Future Work

- Conclusion
  - Specification
  - Solutions to stiffness problem
  - Almost-surely verification of metastable region
- Future Work
  - General solution for stiff system and metastable behaviours
    - An oscillator start-up verification problem
  - Formal specification for more circuits
  - Combine static (symbolic) techniques with reachability computation
- Questions?

Thank You!