Parsimonious, Simulation Based Verification Of Linear Systems

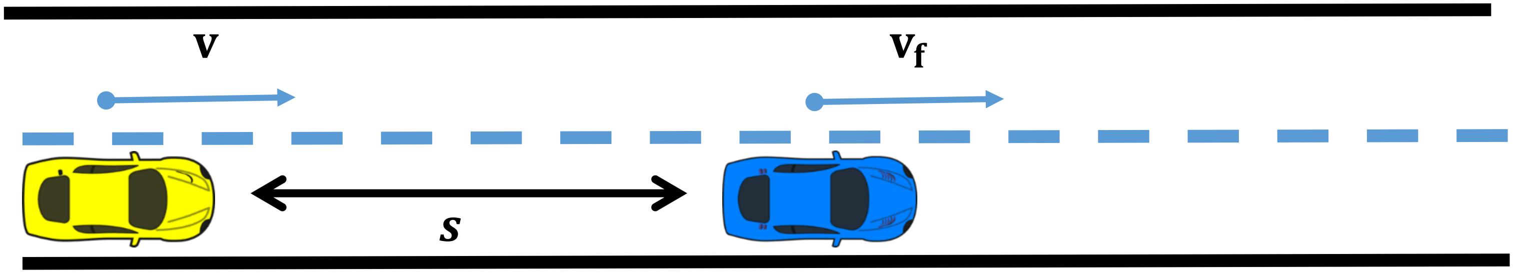
Parasara Sridhar Duggirala

UCONN
UNIVERSITY OF CONNECTICUT

Mahesh Viswanathan

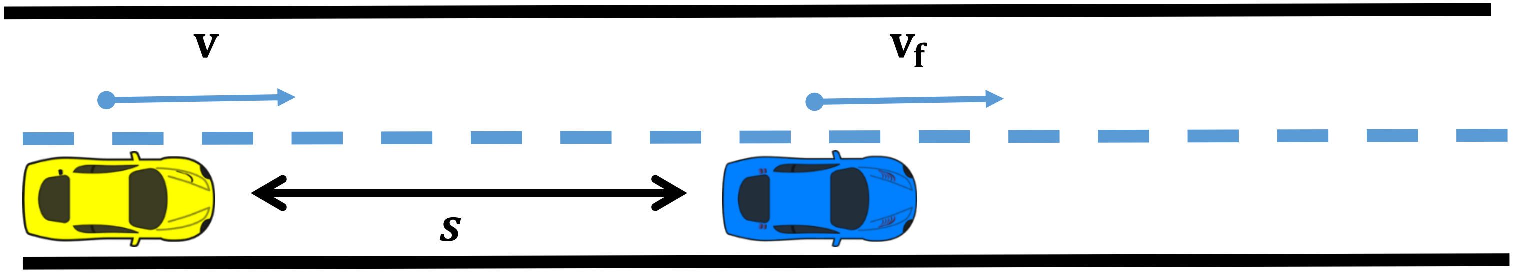
ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Safety Verification: Motivation



- Adaptive cruise control
- Program controller such that $v \rightarrow v_f$ while having $s > limit$.

Safety Verification: Motivation



- Adaptive cruise control
- Program controller such that $v \to v_f$ while having $s > limit$.

ODE model.

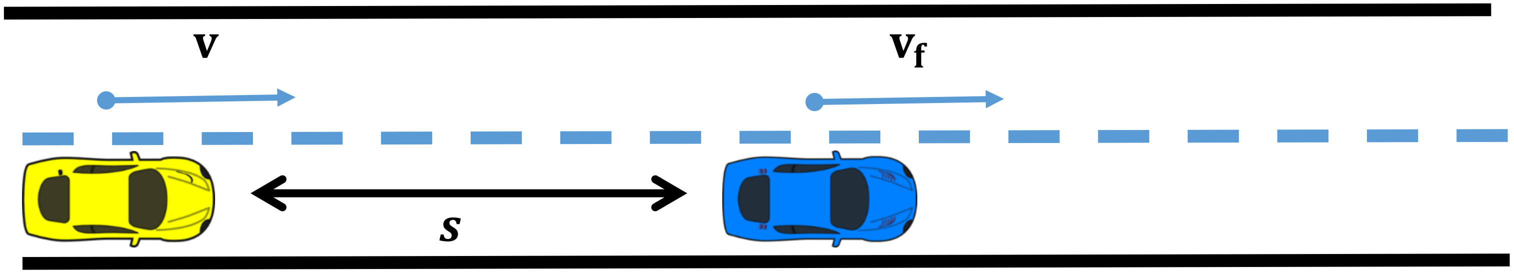
$\dot{s} = v_f - v;$

$\dot{v} = -k_a v + u;$

Controller design

$u = c_1 v + c_2 s + c_3$

Safety Verification: Motivation



- Adaptive cruise control
- Program controller such that $v \rightarrow v_f$ while having $s > limit$.

ODE model.

$\dot{s} = v_f - v;$

$\dot{v} = -k_a v + u;$

Controller design

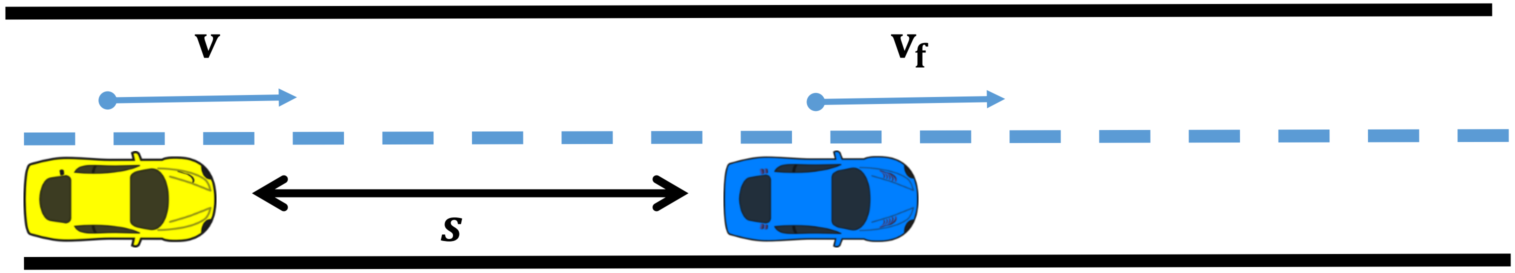
$u = c_1 v + c_2 s + c_3$

Closed loop system

$$\begin{bmatrix} \dot{s} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} s \\ v \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

(represented as $\dot{x} = Ax + B$)

Safety Verification: Motivation



- Adaptive cruise control
- Program controller such that $v \rightarrow v_f$ while having $s > limit$.

ODE model.

$\dot{s} = v_f - v;$

$\dot{v} = -k_a v + u;$

Controller design

$u = c_1 v + c_2 s + c_3$

Closed loop system

$$\begin{bmatrix} \dot{s} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} s \\ v \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

(represented as $\dot{x} = Ax + B$)

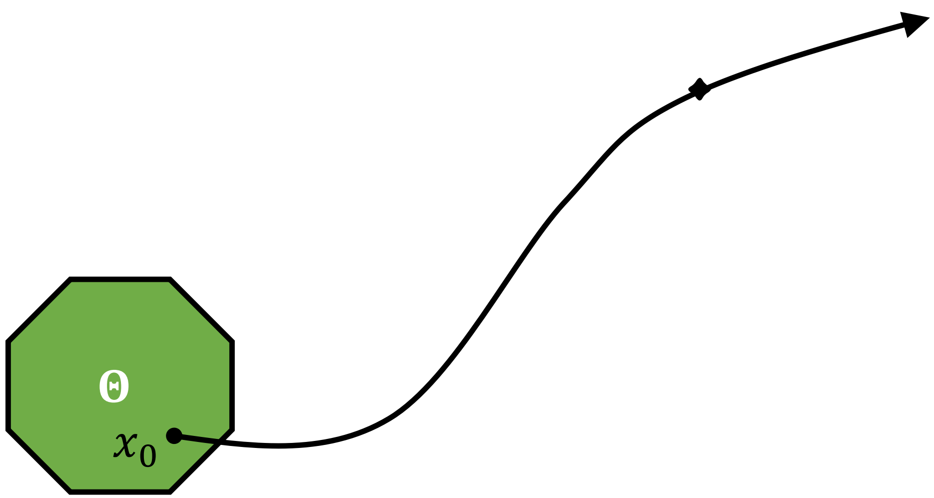
Safety verification problem for linear systems $\dot{x} = Ax + B$

From initial set Θ (dis)prove that no trajectory enters the unsafe set U

Solution: Reachable Set

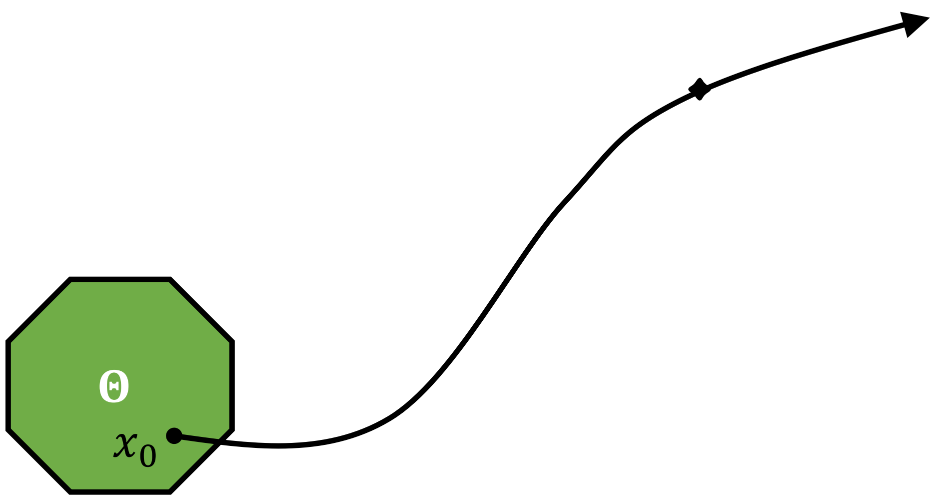
System: $\dot{x} = Ax + B$, initial set Θ (polyhedra), unsafe set U.

$$\xi(x_0, t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)}Bd\tau$$



Solution: Reachable Set

System: $\dot{x} = Ax + B$, initial set Θ (polyhedra), unsafe set U.



$$\xi(x_0, t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)} B d\tau$$

Procedure to compute reachable set

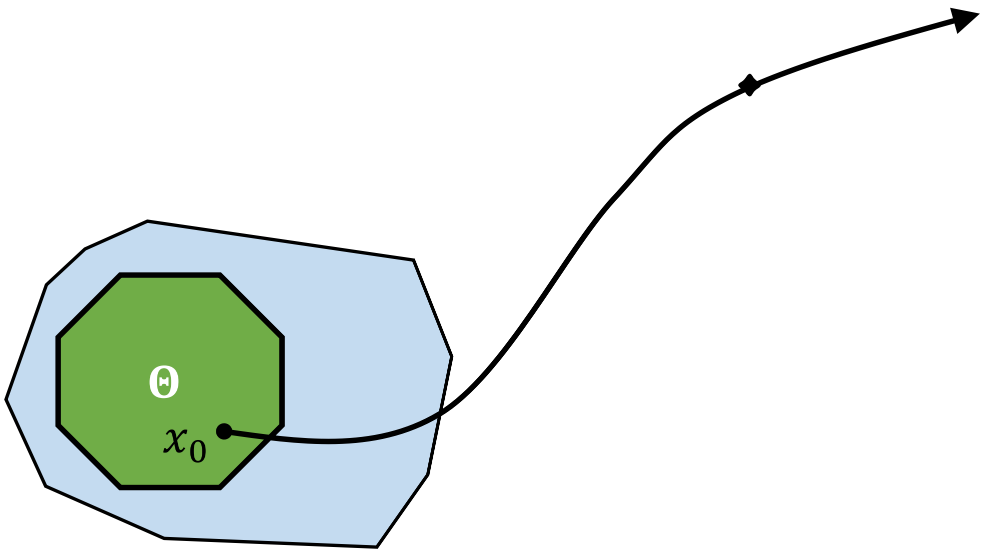
1. Represent the set Θ using data structure

Data structure
SpaceEx – Support Functions
CORA – Zonotopes
Flow* – Taylor Models

Solution: Reachable Set

System: $\dot{x} = Ax + B$, initial set Θ (polyhedra), unsafe set U.

$$\xi(x_0, t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)} B d\tau$$



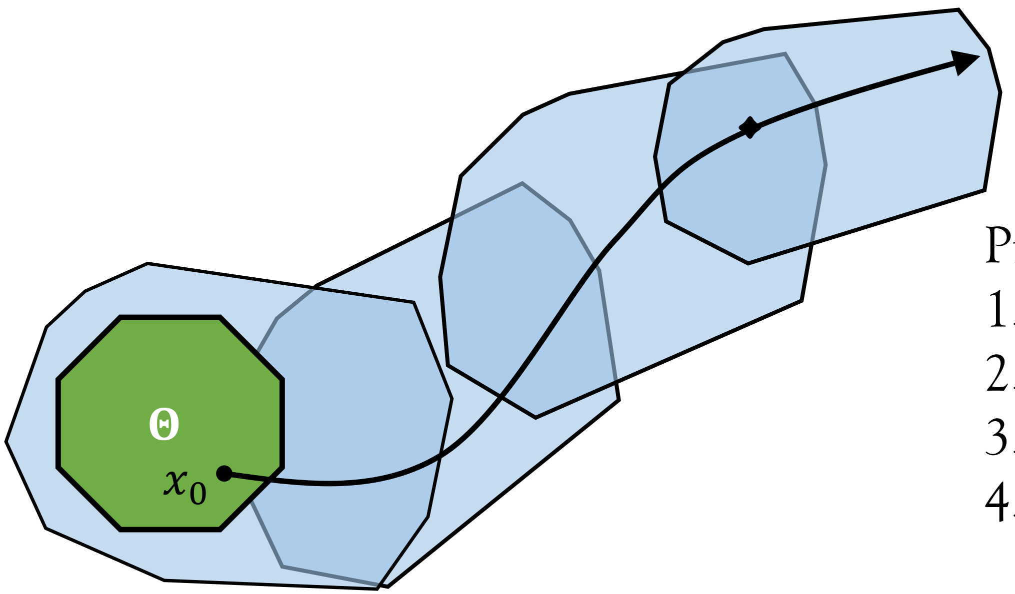
Procedure to compute reachable set

1. Represent the set Θ using data structure
2. Select a time interval h.
3. Compute $Post(\Theta, h)$ for $[0, h]$

Data structure
SpaceEx – Support Functions
CORA – Zonotopes
Flow* – Taylor Models

Solution: Reachable Set

System: $\dot{x} = Ax + B$, initial set Θ (polyhedra), unsafe set U.



$$\xi(x_0, t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)}Bd\tau$$

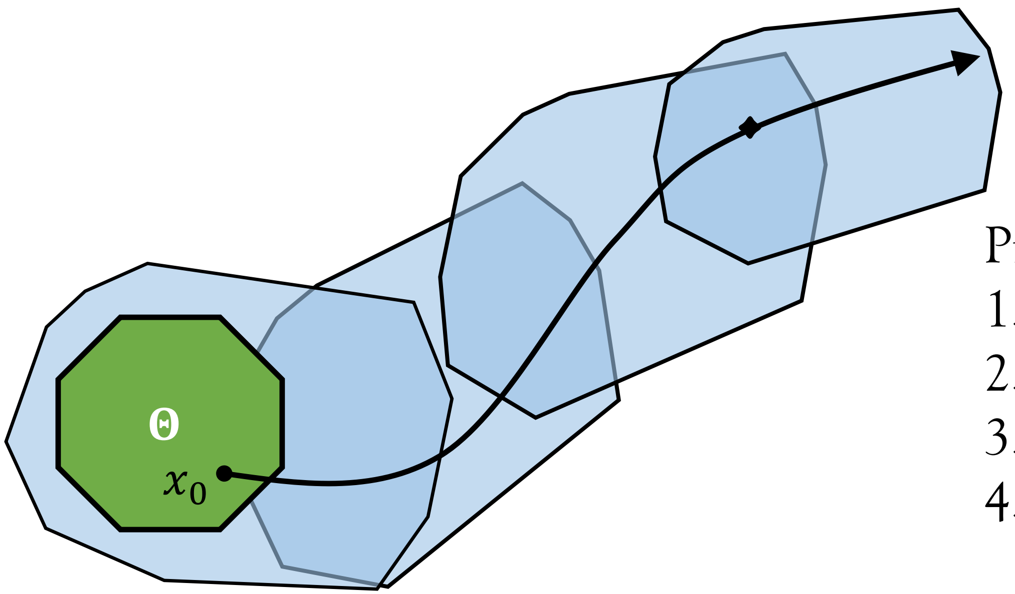
Procedure to compute reachable set

1. Represent the set Θ using data structure
2. Select a time interval h.
3. Compute $Post(\Theta, h)$ for $[0, h]$
4. Iterate for future intervals.

Data structure
SpaceEx – Support Functions
CORA – Zonotopes
Flow* – Taylor Models

Solution: Reachable Set

System: $\dot{x} = Ax + B$, initial set Θ (polyhedra), unsafe set U.



$$\xi(x_0, t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)}Bd\tau$$

Procedure to compute reachable set

1. Represent the set Θ using data structure
2. Select a time interval h.
3. Compute $Post(\Theta, h)$ for $[0, h]$
4. Iterate for future intervals.

Data structure
SpaceEx – Support Functions
CORA – Zonotopes
Flow* – Taylor Models

Drawbacks

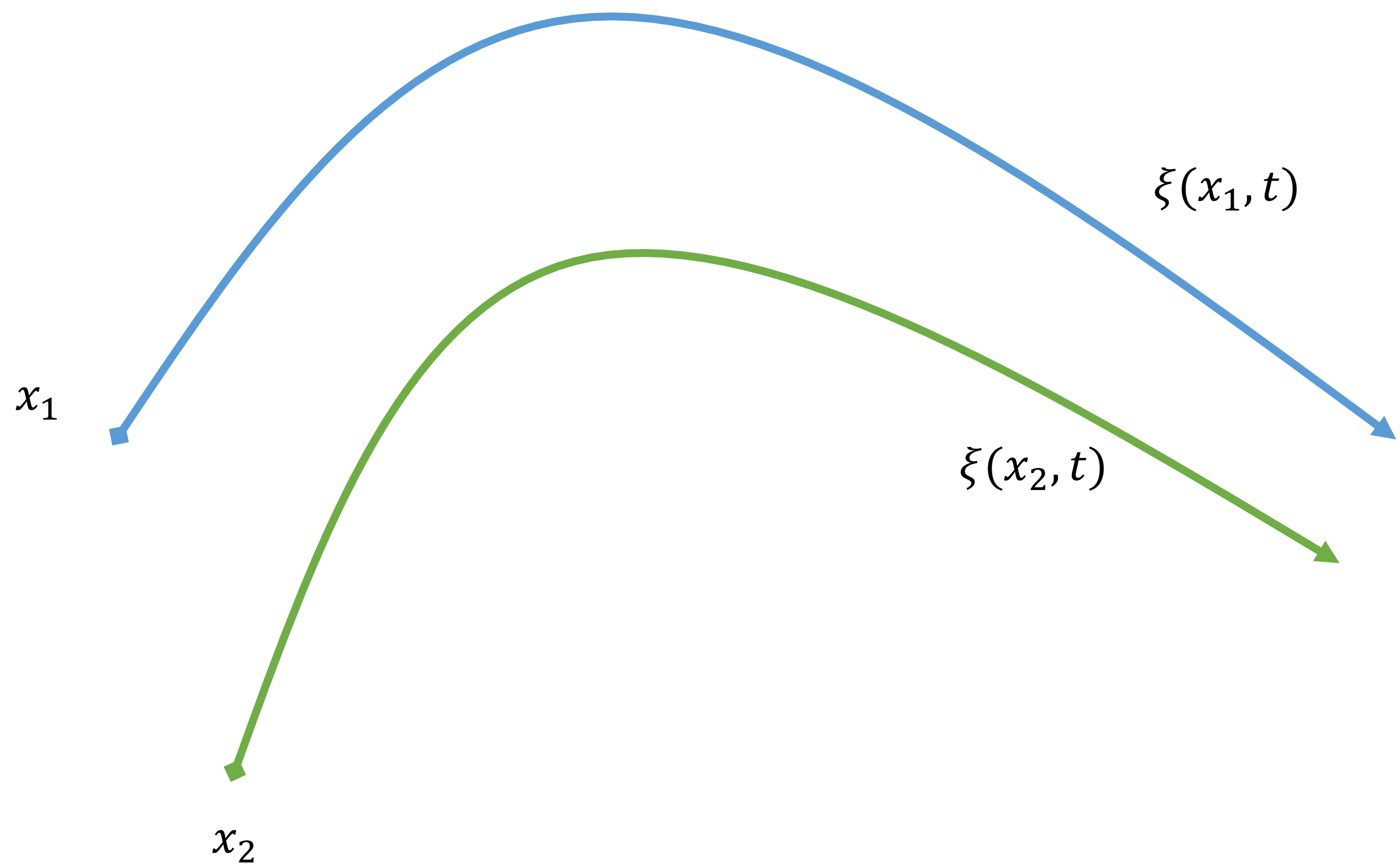
1. Representation cost grows with n
2. Only overapproximation
3. Cannot be directly applied for time varying linear systems

This Paper: Contributions

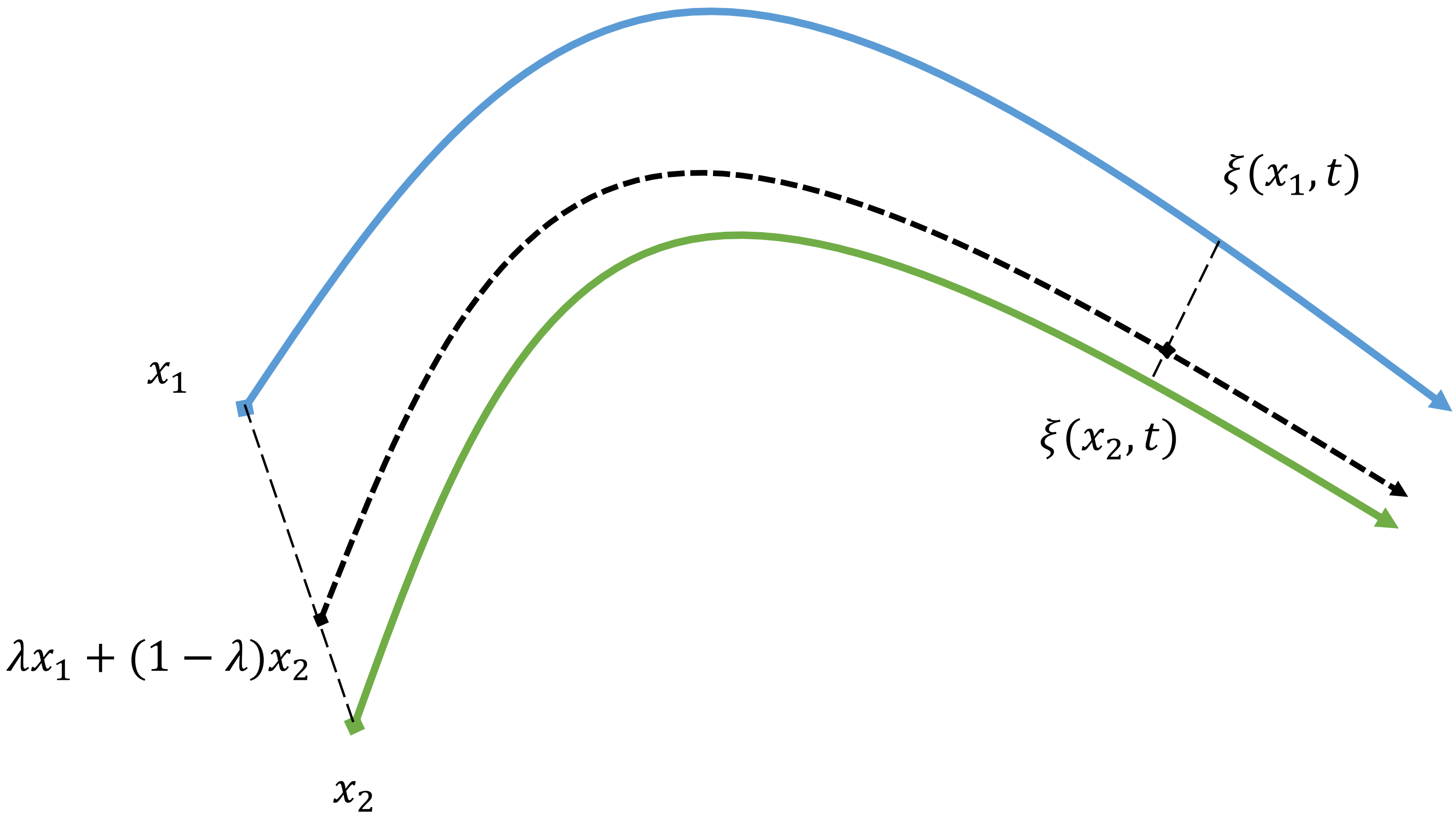
New simulation based verification for linear systems.

1. **<u>For n-dimensional system, $n + 1$ simulations suffice.</u>**
2. Works for both time invariant and time variant systems.
3. Works for non-convex and unbounded initial set.
4. Can compute over- and under-approximation.

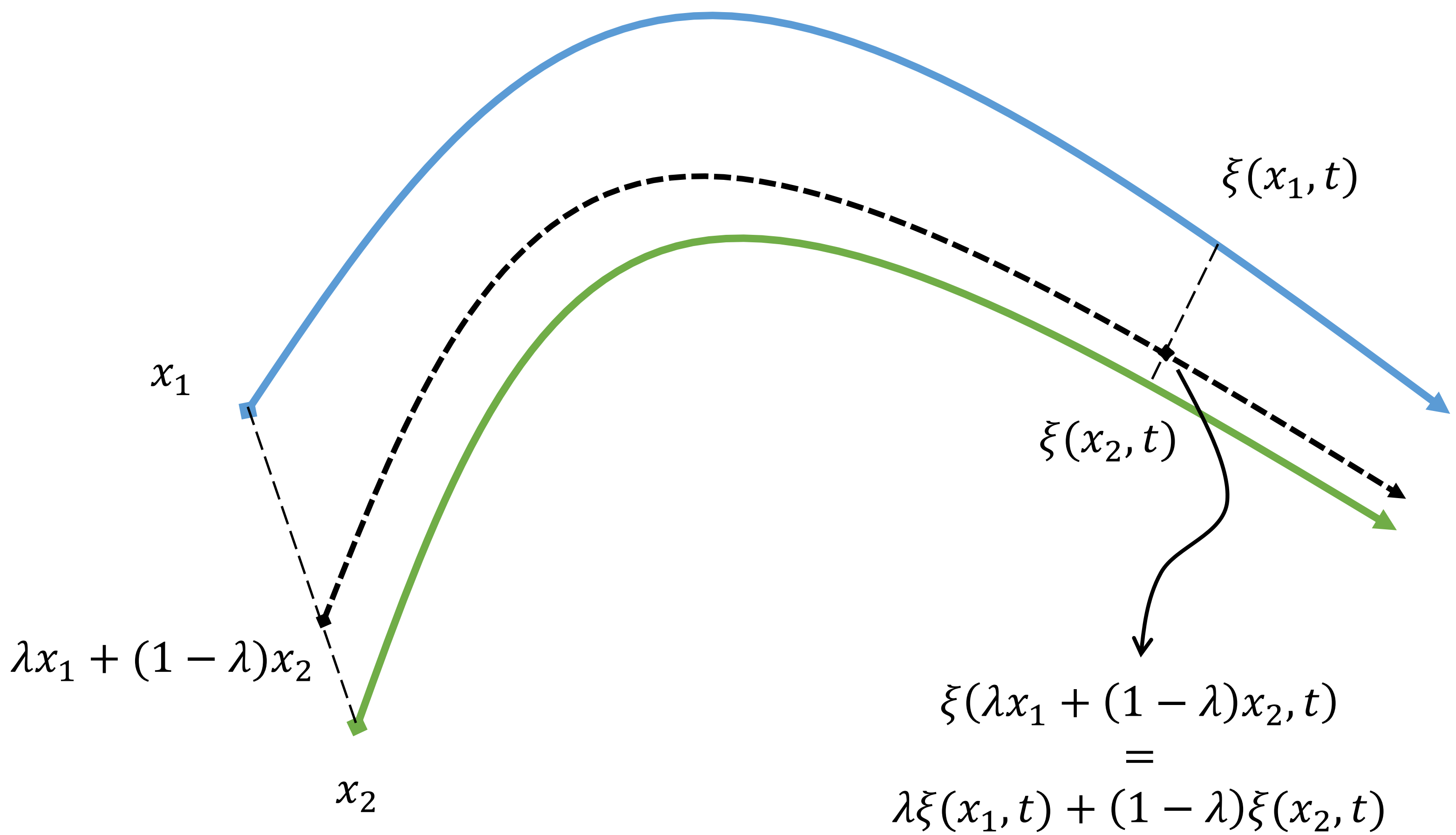
The How: Superposition Principle



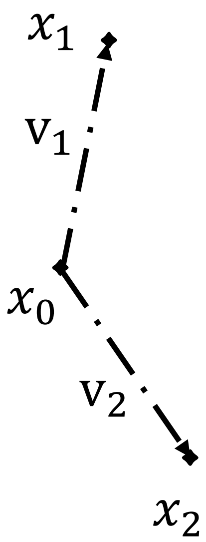
The How: Superposition Principle



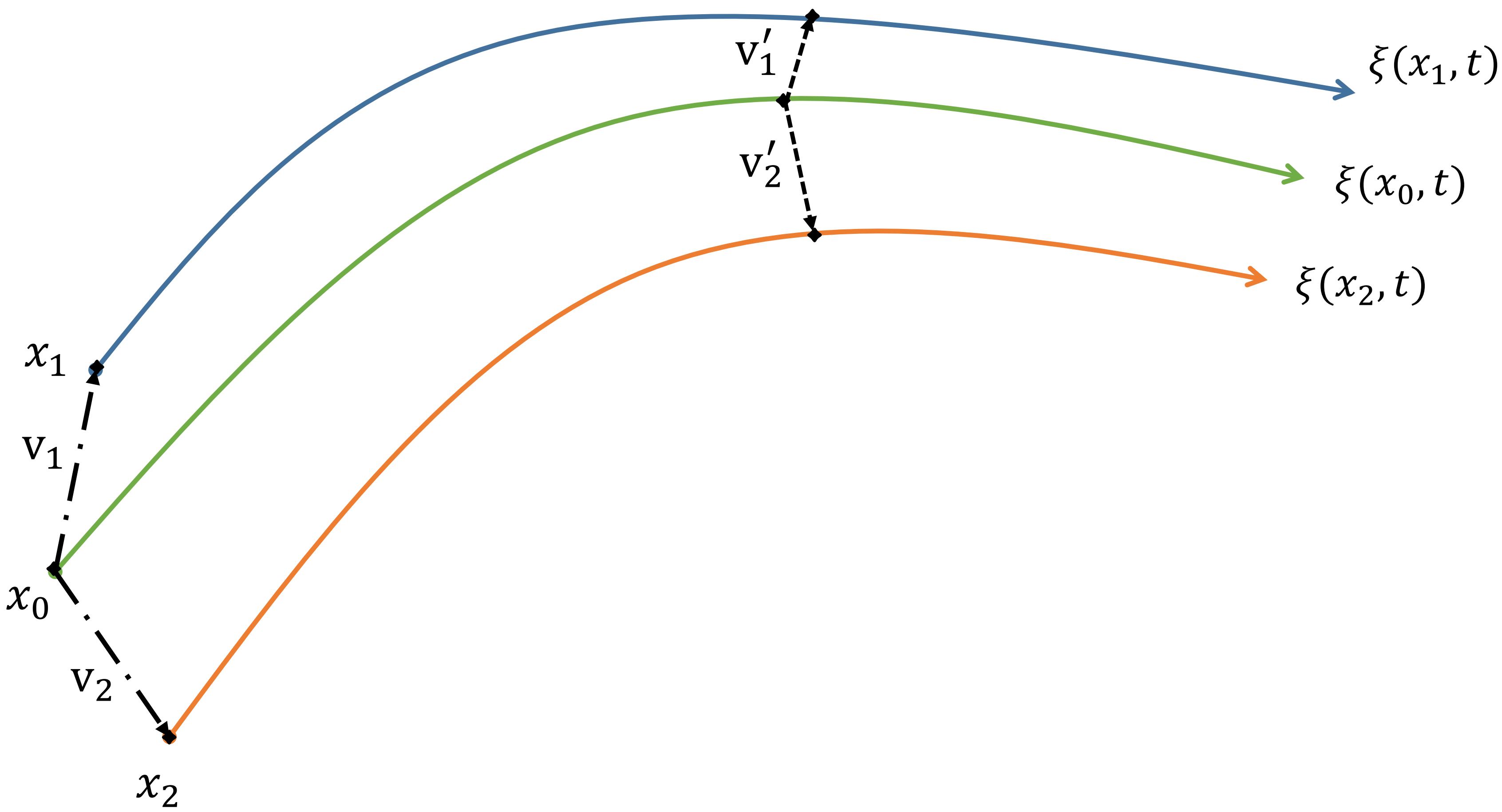
The How: Superposition Principle



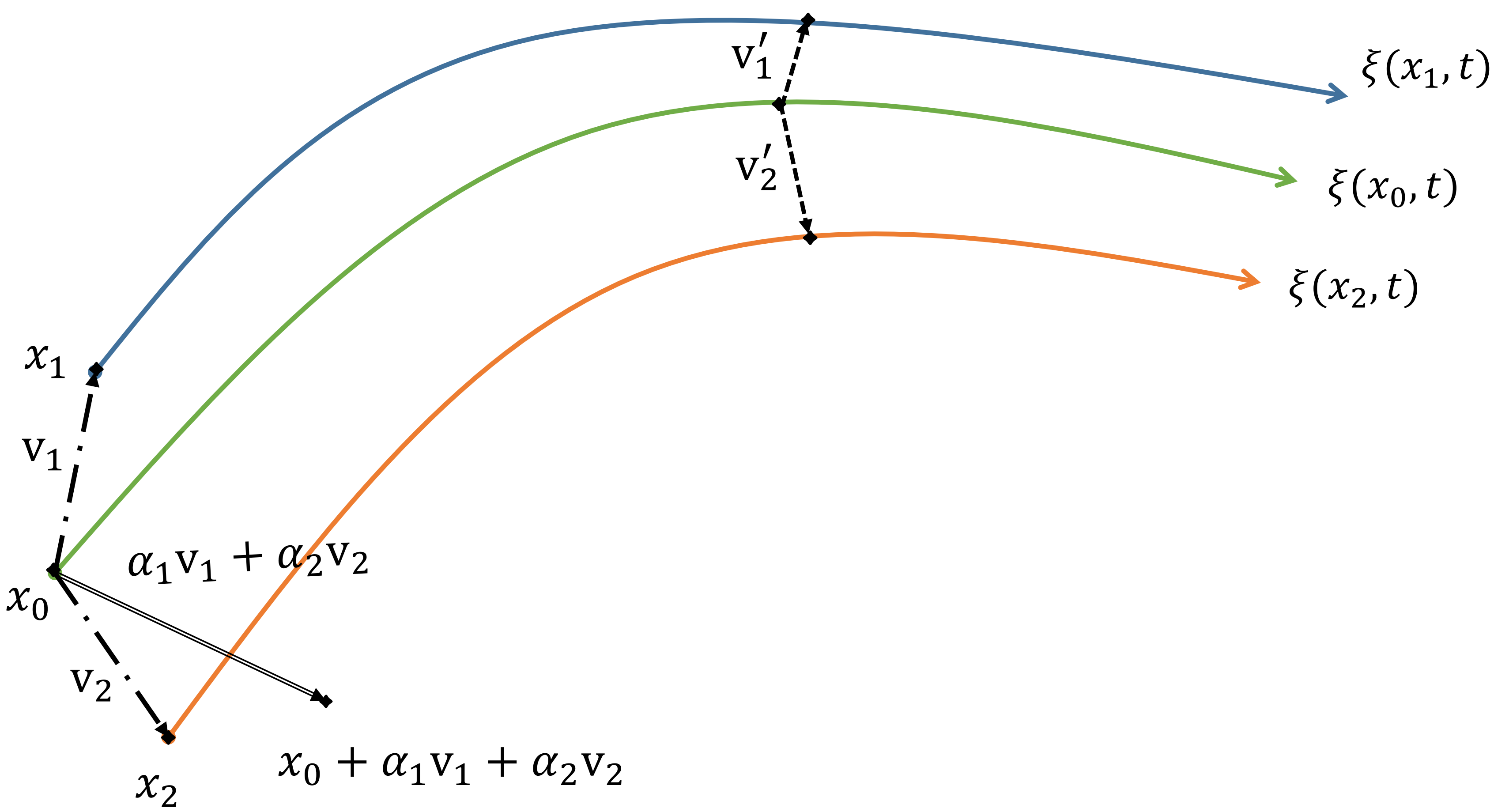
Implications Of Superposition



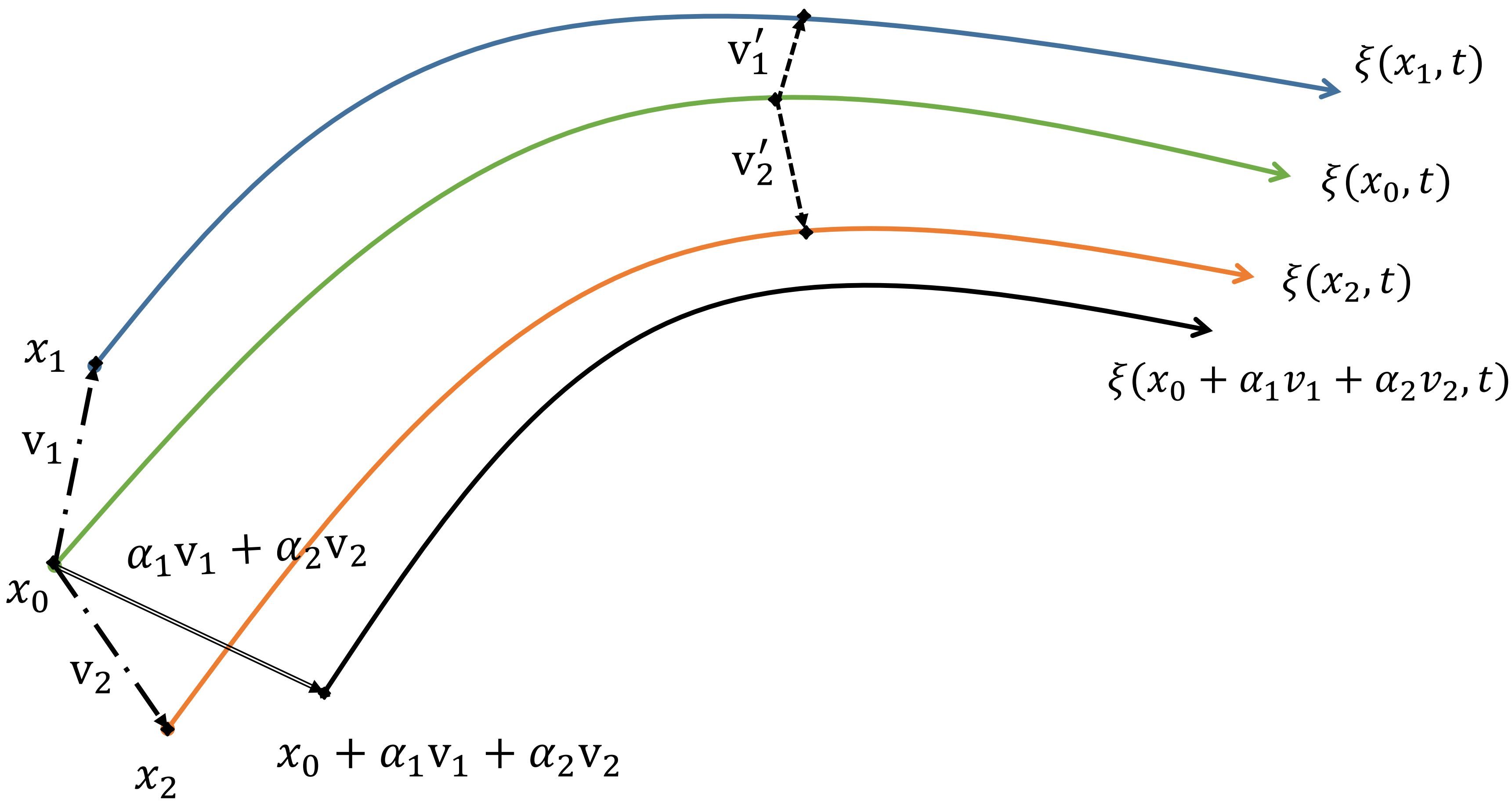
Implications Of Superposition



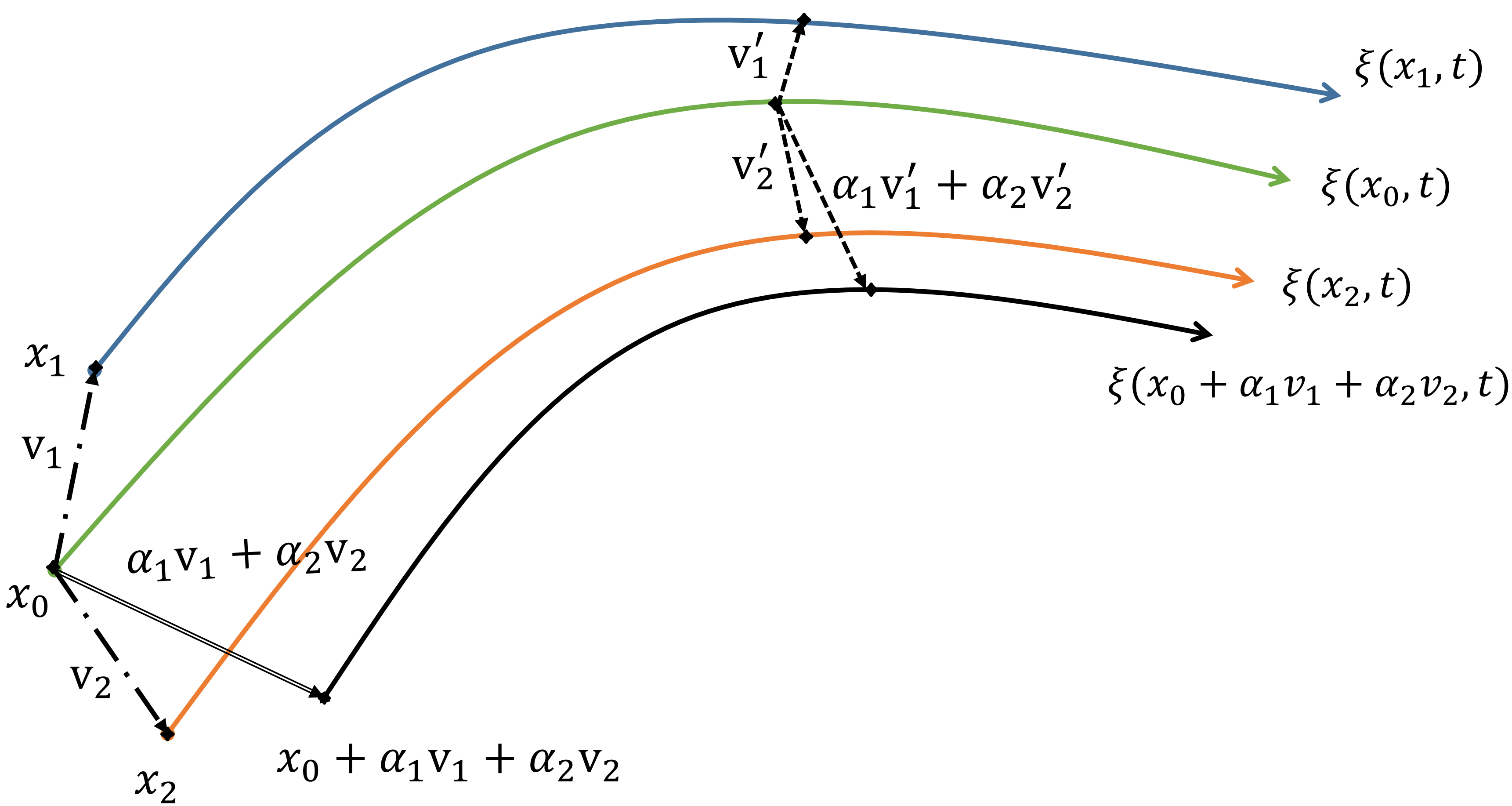
Implications Of Superposition



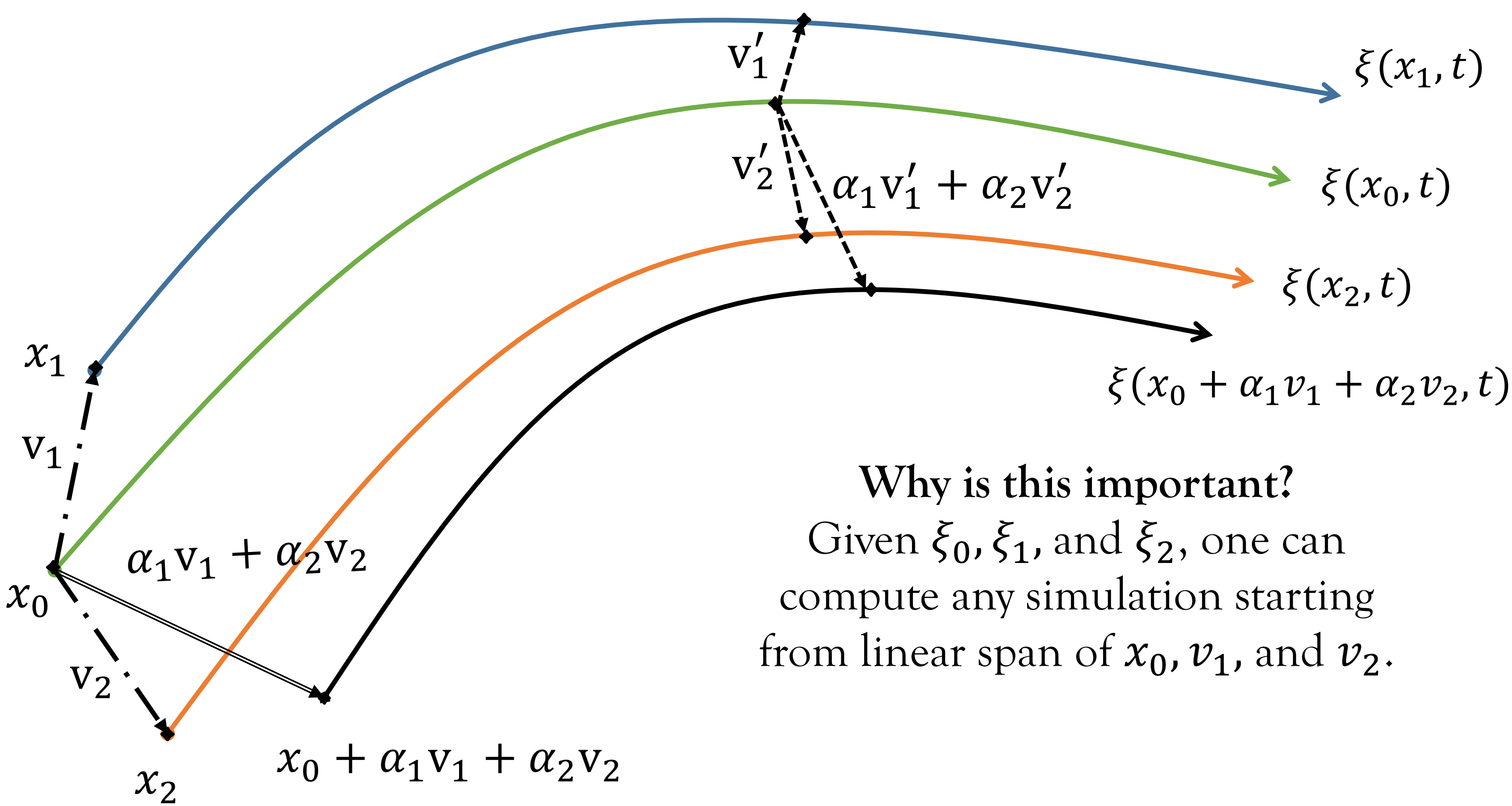
Implications Of Superposition



Implications Of Superposition



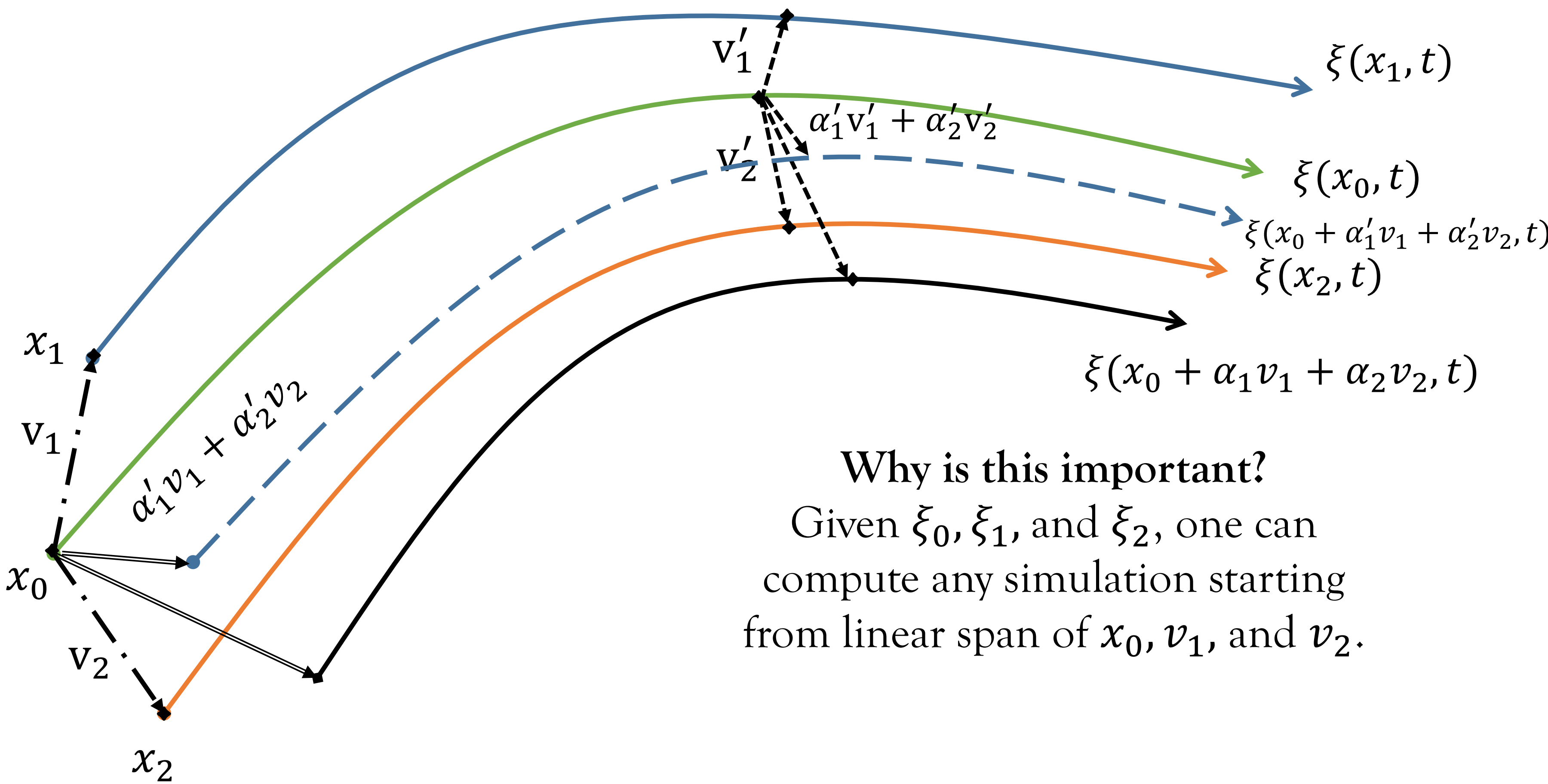
Implications Of Superposition



Why is this important?

Given ξ_0, ξ_1, and ξ_2, one can compute any simulation starting from linear span of x_0, v_1, and v_2.

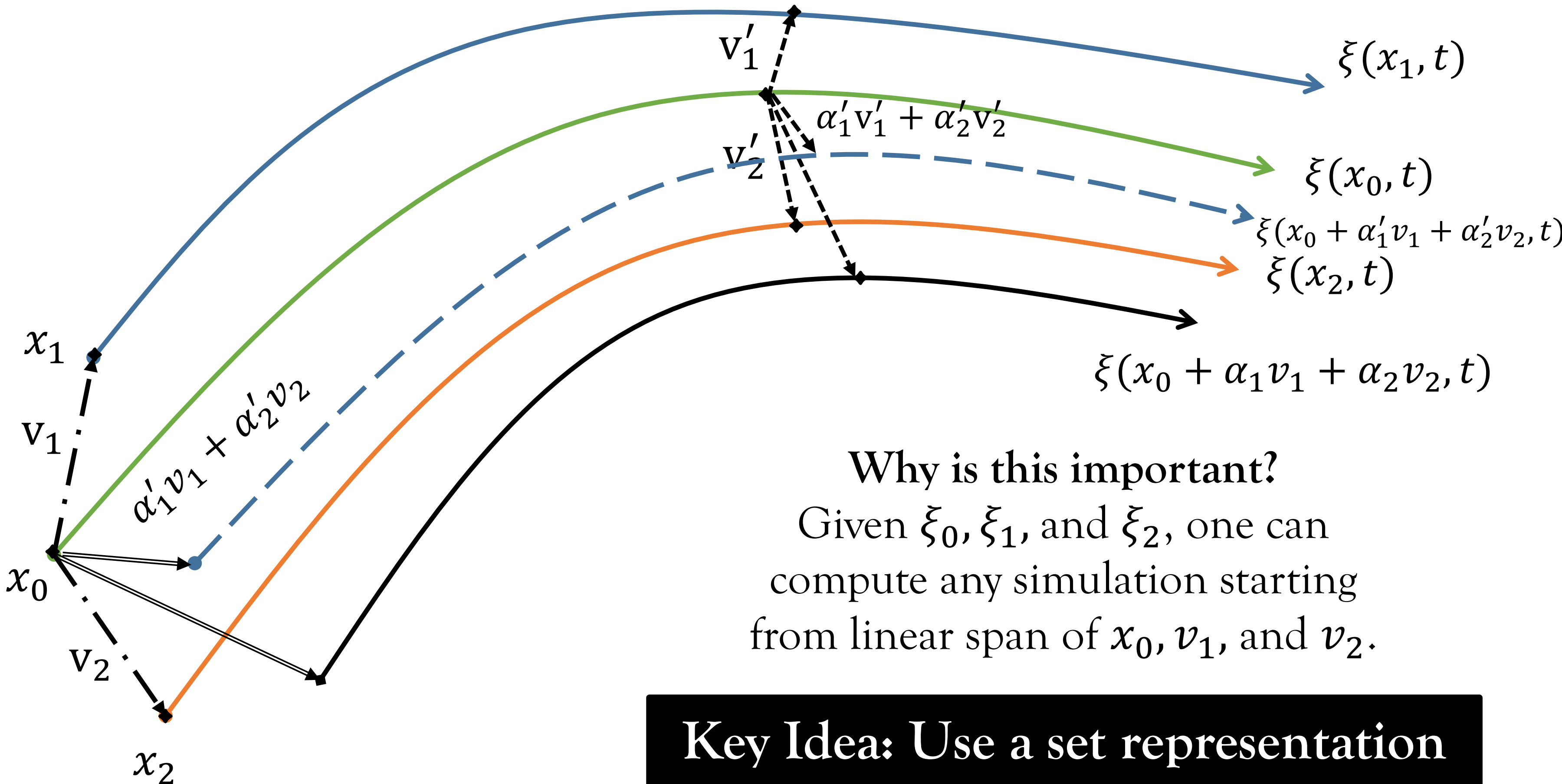
Implications Of Superposition



Why is this important?

Given ξ_0, ξ_1, and ξ_2, one can compute any simulation starting from linear span of x_0, v_1, and v_2.

Implications Of Superposition



Why is this important?

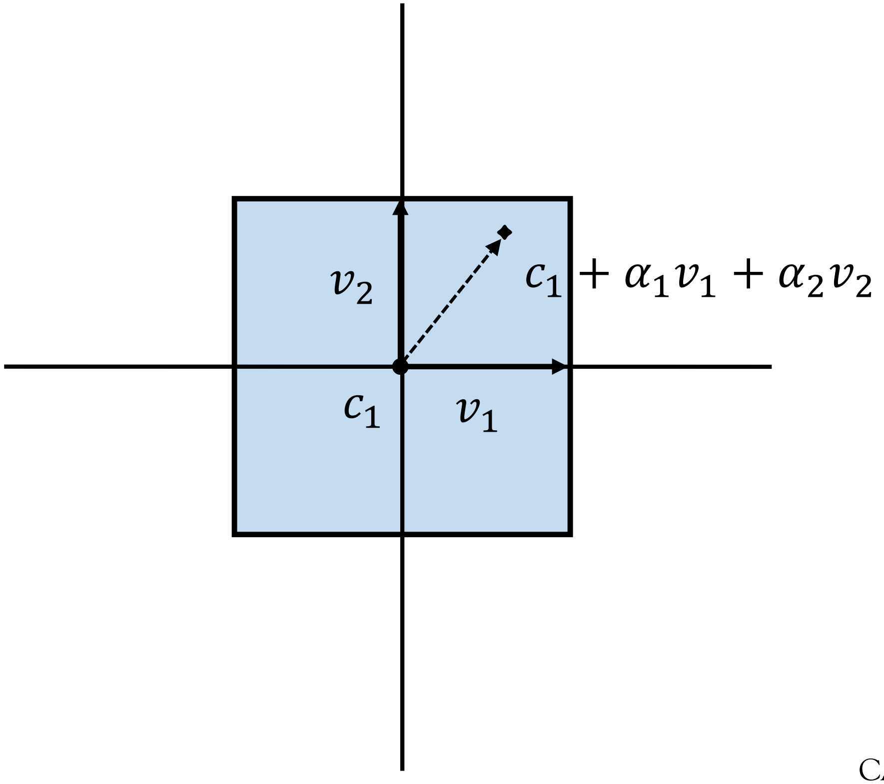
Given ξ_0, ξ_1, and ξ_2, one can compute any simulation starting from linear span of x_0, v_1, and v_2.

Key Idea: Use a set representation that exploits this property

Representation: Generalized Stars

- Generalized star is represented as $\langle c, V, P \rangle$
- c – center, V – set of vectors, P – predicate.

$$\langle c, V, P \rangle = \{\, x \mid \exists \bar{\alpha} = (\alpha_1, \dots, \alpha_n), \mathrm{c} + \Sigma_i \alpha_i v_i = x, P(\bar{\alpha}) = \top \}$$

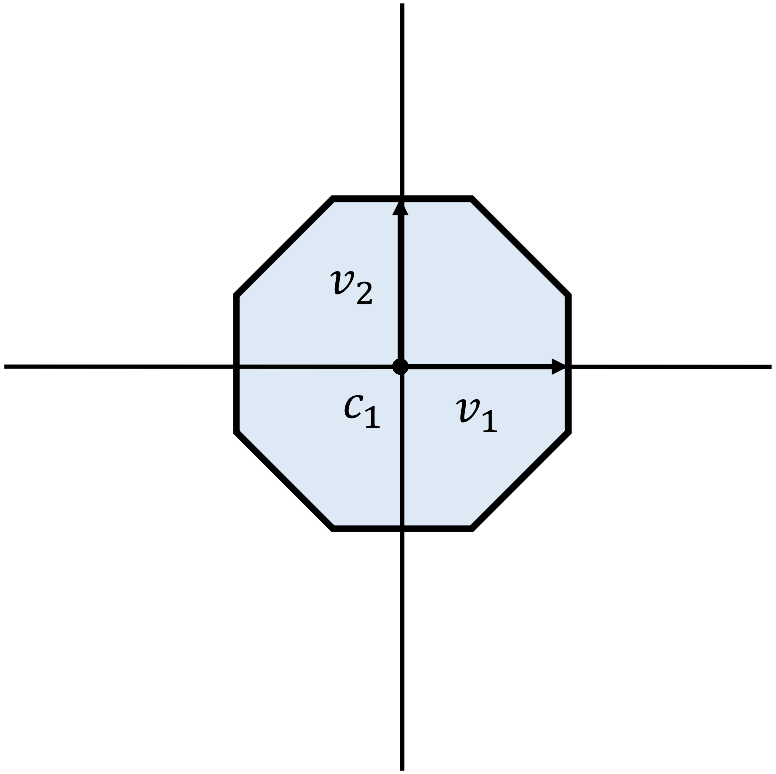


$$P(\langle \alpha_1, \alpha_2 \rangle) \triangleq |\alpha_1| \leq 1 \wedge |\alpha_2| \leq 1$$

Representation: Generalized Stars

- Generalized star is represented as $\langle c, V, P\rangle$
- c – center, V – set of vectors, P – predicate.

$$\langle c, V, P\rangle = \{\, x \mid \exists \bar{\alpha} = (\alpha_1, \ldots, \alpha_n), \mathrm{c} + \Sigma_i \alpha_i v_i = x, P(\bar{\alpha}) = \top\}$$

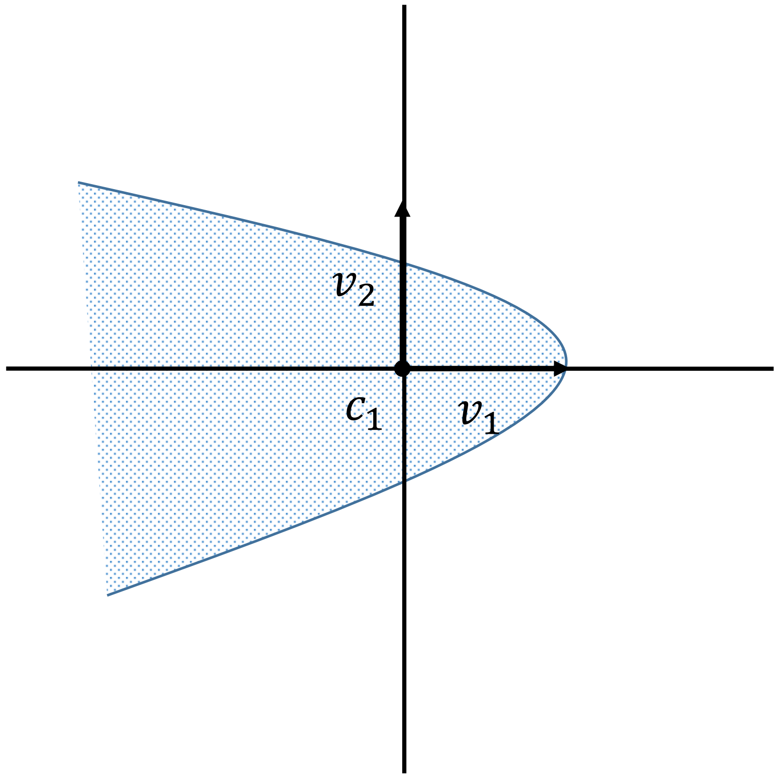


$$P(\langle \alpha_1, \alpha_2\rangle) \triangleq |\alpha_1| \le 1 \wedge |\alpha_2| \le 1 \wedge |\alpha_1 + \alpha_2| \le 1.5$$

Representation: Generalized Stars

- Generalized star is represented as $\langle c, V, P\rangle$
- c – center, V – set of vectors, P – predicate.

$$\langle c, V, P\rangle = \{\, x \mid \exists\bar{\alpha} = (\alpha_1, \ldots, \alpha_n), \mathrm{c} + \Sigma_i \alpha_i v_i = x, P(\bar{\alpha}) = \top\}$$

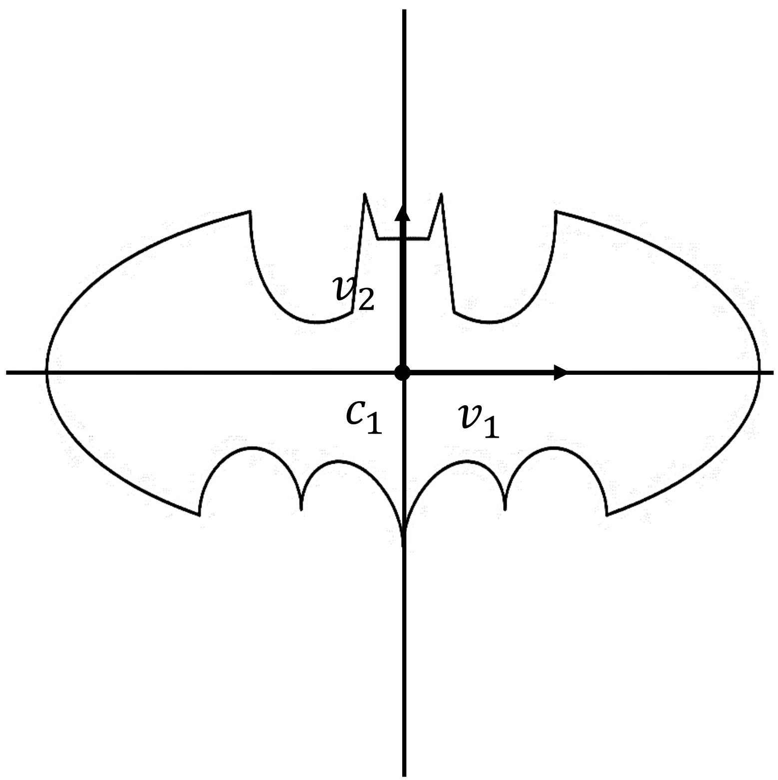


$$P(\langle\alpha_1, \alpha_2\rangle) \triangleq \alpha_1 \leq 1 - \alpha_2^2$$

Representation: Generalized Stars

- Generalized star is represented as $\langle c, V, P \rangle$
- c – center, V – set of vectors, P – predicate.

$$\langle c, V, P \rangle = \{\, x \mid \exists \bar{\alpha} = (\alpha_1, \dots, \alpha_n), \mathrm{c} + \Sigma_i \alpha_i v_i = x, P(\bar{\alpha}) = \top \}$$



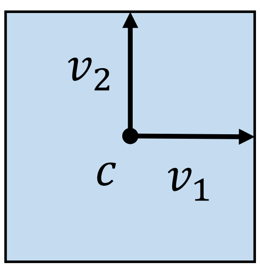
$$P(\langle \alpha_1, \alpha_2 \rangle)$$

$$\triangleq$$

$$1.5 * sqrt\left(\left(-abs(abs(x) - 1)\right) * \frac{abs(3 - abs(x))}{(abs(x) - 1) * (3 - abs(x))}\right) * \left(1 + \frac{abs(abs(x) - 3)}{abs(x) - 3}\right) * sqrt\left(1 - \left(\frac{x}{7}\right)^2\right) +$$
$$\left(4.5 + 0.75 * \left(abs(x - 0.5) + abs(x + 0.5)\right) - 2.75 * \left(abs(x - 0.75) + abs(x + 0.75)\right)\right) * \left(1 + \frac{abs(1 - abs(x))}{1 - abs(x)}\right),$$
$$(-3) * sqrt\left(1 - \left(\frac{x}{7}\right)^2\right) * sqrt\left(\frac{abs(abs(x) - 4)}{abs(x) - 4}\right), abs\left(\frac{x}{2}\right) - 0.0913722 * x^2 - 3 + sqrt\left(1 - (abs(abs(x) - 2) - 1)^2\right),$$
$$(2.71052 + 1.5 - 0.5 * abs(x) - 1.35526 * sqrt(4 - (abs(x) - 1)\hat{}2)) * sqrt(abs(abs(x) - 1)/(abs(x) - 1))$$

Reachable Set Computation Using Simulations For Generalized Stars

Given $\Theta \triangleq \langle c, V, P \rangle$ to compute reachable set



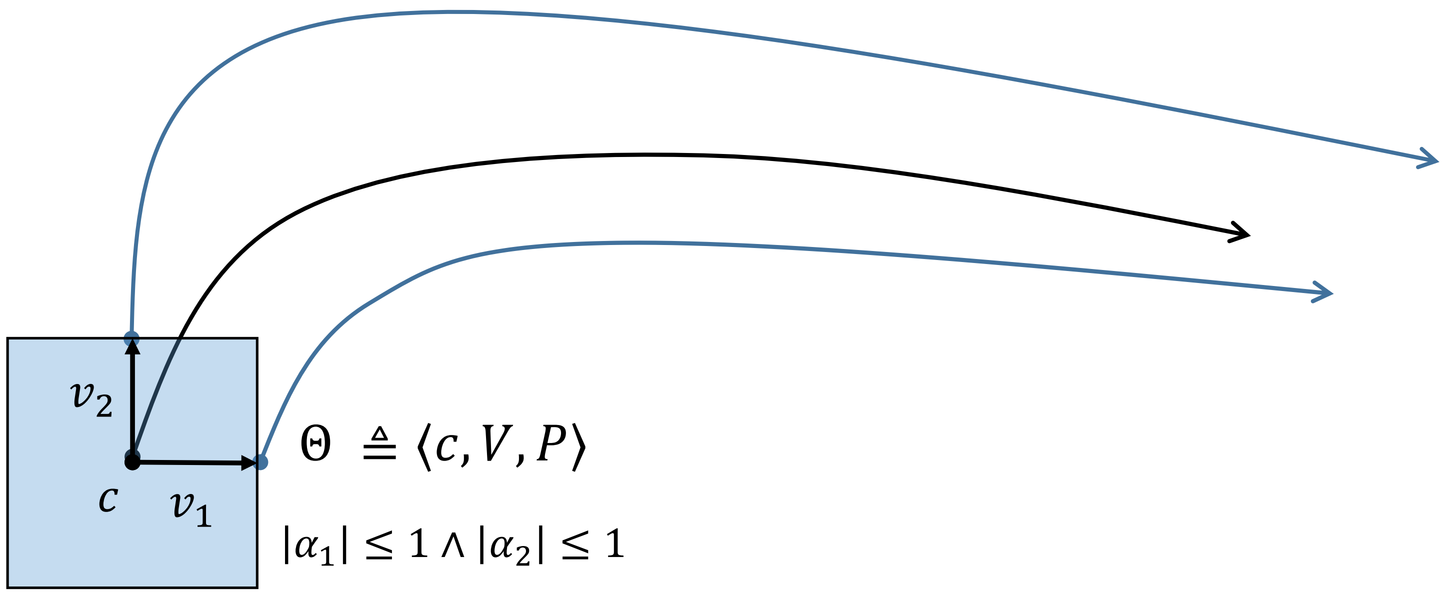
$\Theta \triangleq \langle c, V, P \rangle$

$|\alpha_1| \leq 1 \wedge |\alpha_2| \leq 1$

Reachable Set Computation Using Simulations For Generalized Stars

Given $\Theta \triangleq \langle c, V, P \rangle$ to compute reachable set

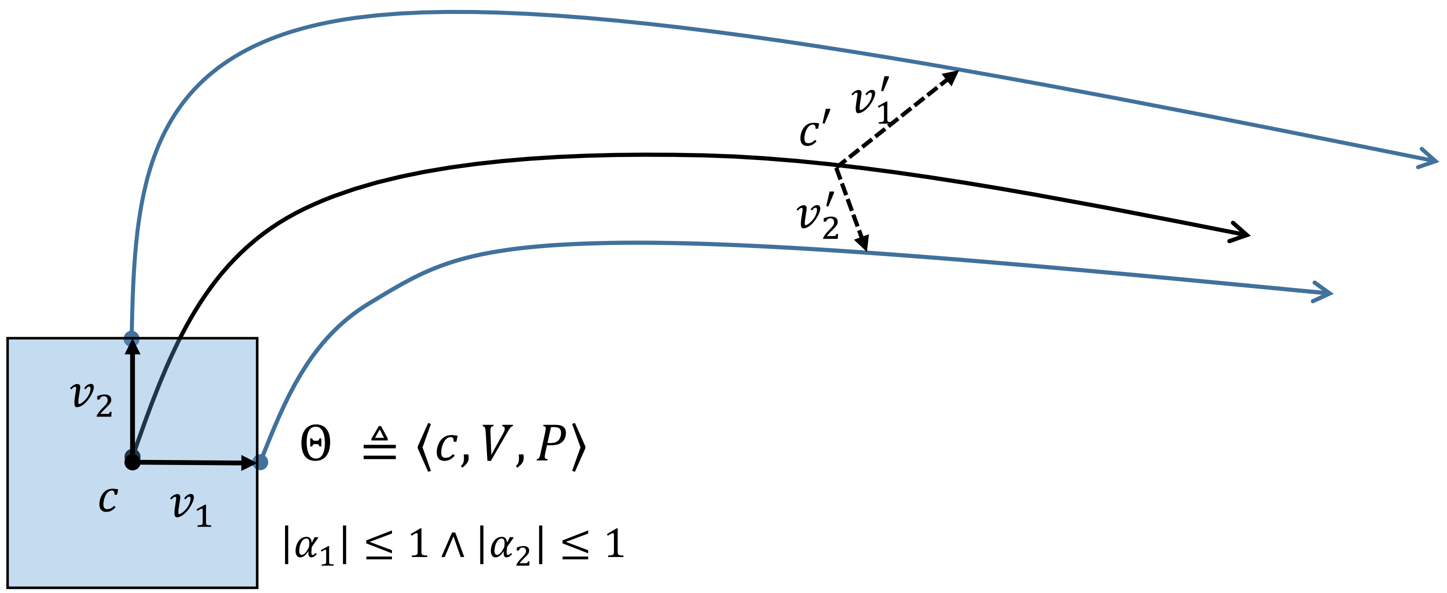
1. Simulate from c
2. Simulate from $c + v_i$ for each i



Reachable Set Computation Using Simulations For Generalized Stars

Given $\Theta \triangleq \langle c, V, P \rangle$ to compute reachable set

1. Simulate from c
2. Simulate from $c + v_i$ for each i



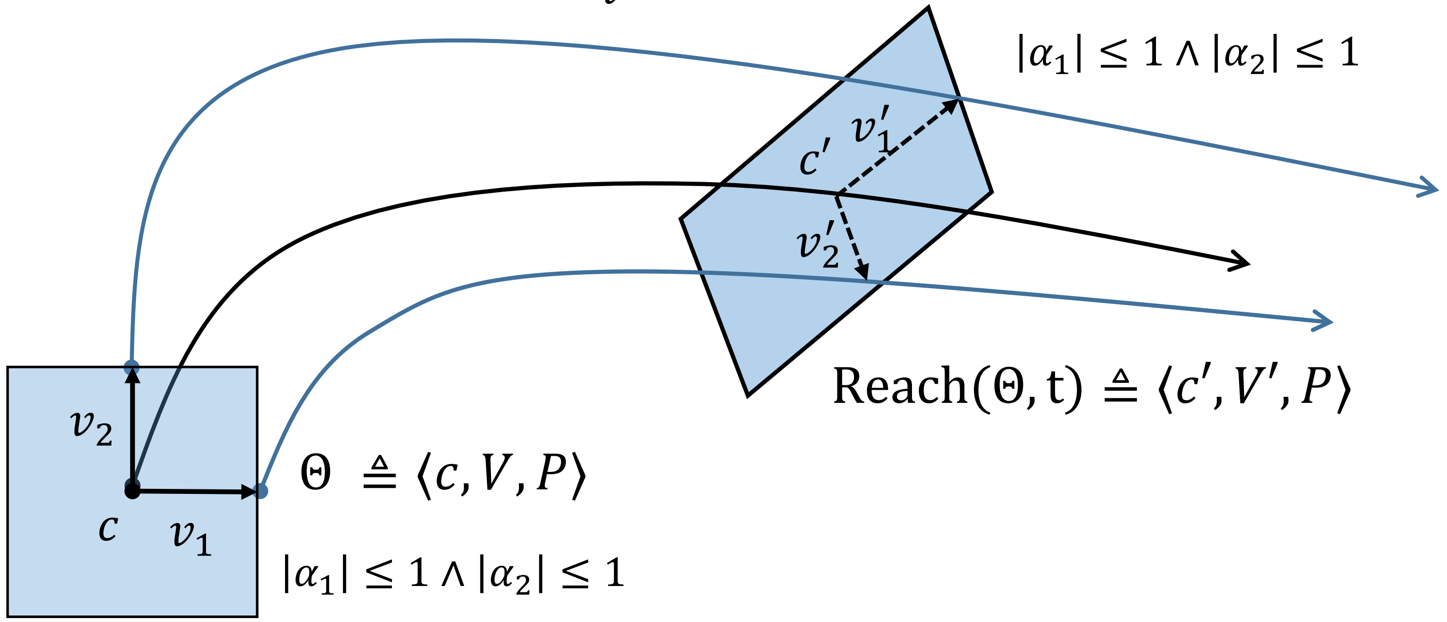
Reachable set at time t is given by $\langle c', V', P \rangle$ where

1. c' is the simulation corresponding to c
2. $v_i{}'$ is the difference of simulations from $c + v_i$ and from c

Reachable Set Computation Using Simulations For Generalized Stars

Given $\Theta \triangleq \langle c, V, P \rangle$ to compute reachable set

1. Simulate from c
2. Simulate from $c + v_i$ for each i



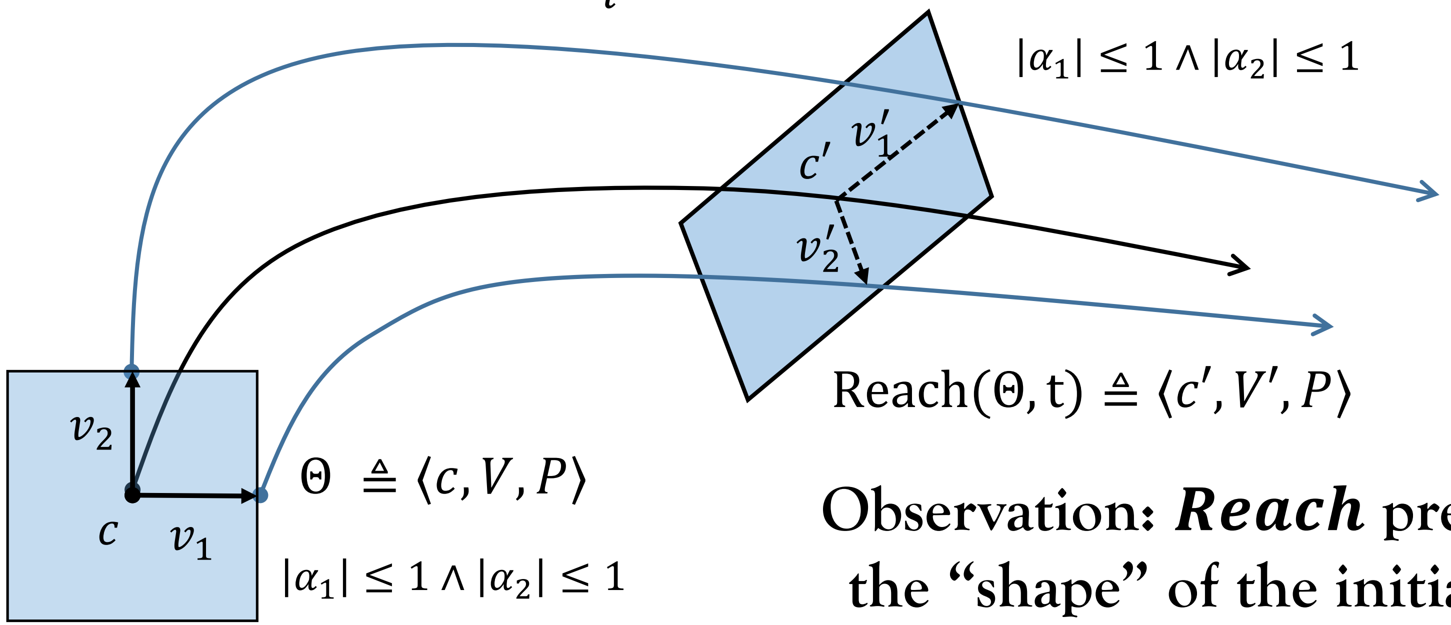
Reachable set at time t is given by $\langle c', V', P \rangle$ where

1. c' is the simulation corresponding to c
2. ${v_i}'$ is the difference of simulations from $c + v_i$ and from c

Reachable Set Computation Using Simulations For Generalized Stars

Given $\Theta \triangleq \langle c, V, P \rangle$ to compute reachable set

1. Simulate from c
2. Simulate from $c + v_i$ for each i



Observation: *Reach* preserves the "shape" of the initial set.

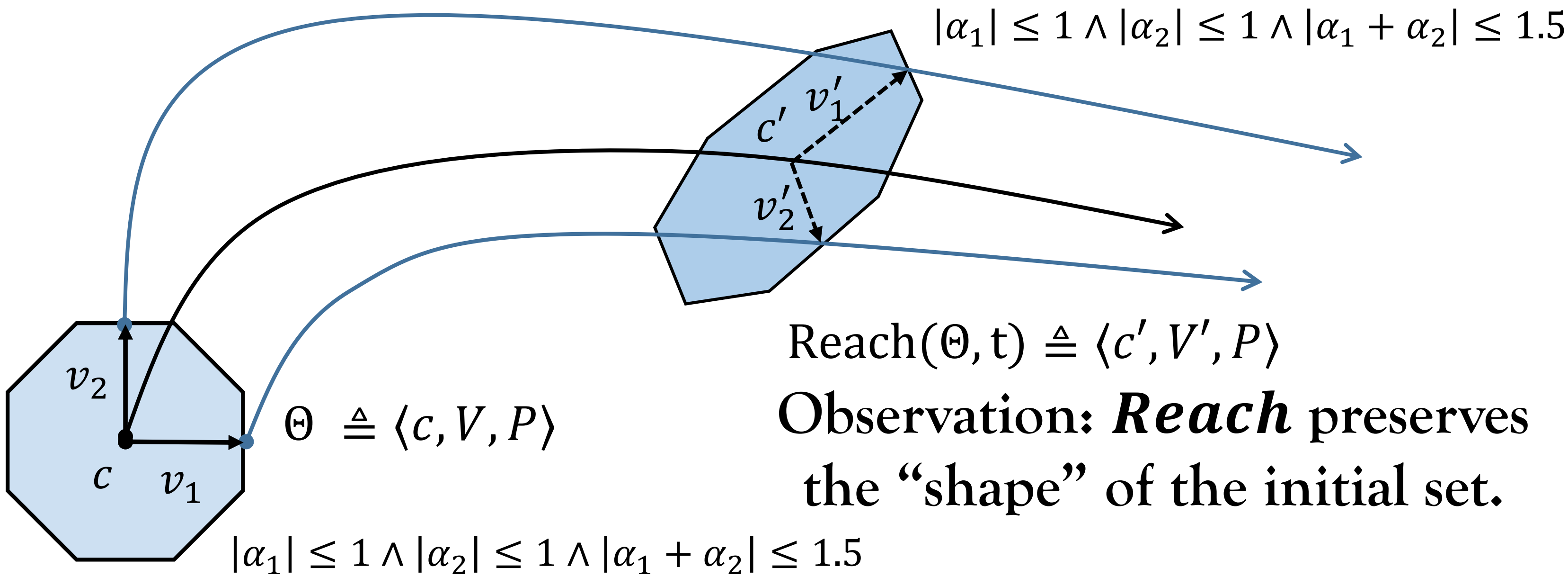
Reachable set at time t is given by $\langle c', V', P \rangle$ where

1. c' is the simulation corresponding to c
2. $v_i{}'$ is the difference of simulations from $c + v_i$ and from c

Reachable Set Computation Using Simulations For Generalized Stars

Given $\Theta \triangleq \langle c, V, P \rangle$ to compute reachable set

1. Simulate from c
2. Simulate from $c + v_i$ for each i



Observation: *Reach* preserves the “shape” of the initial set.

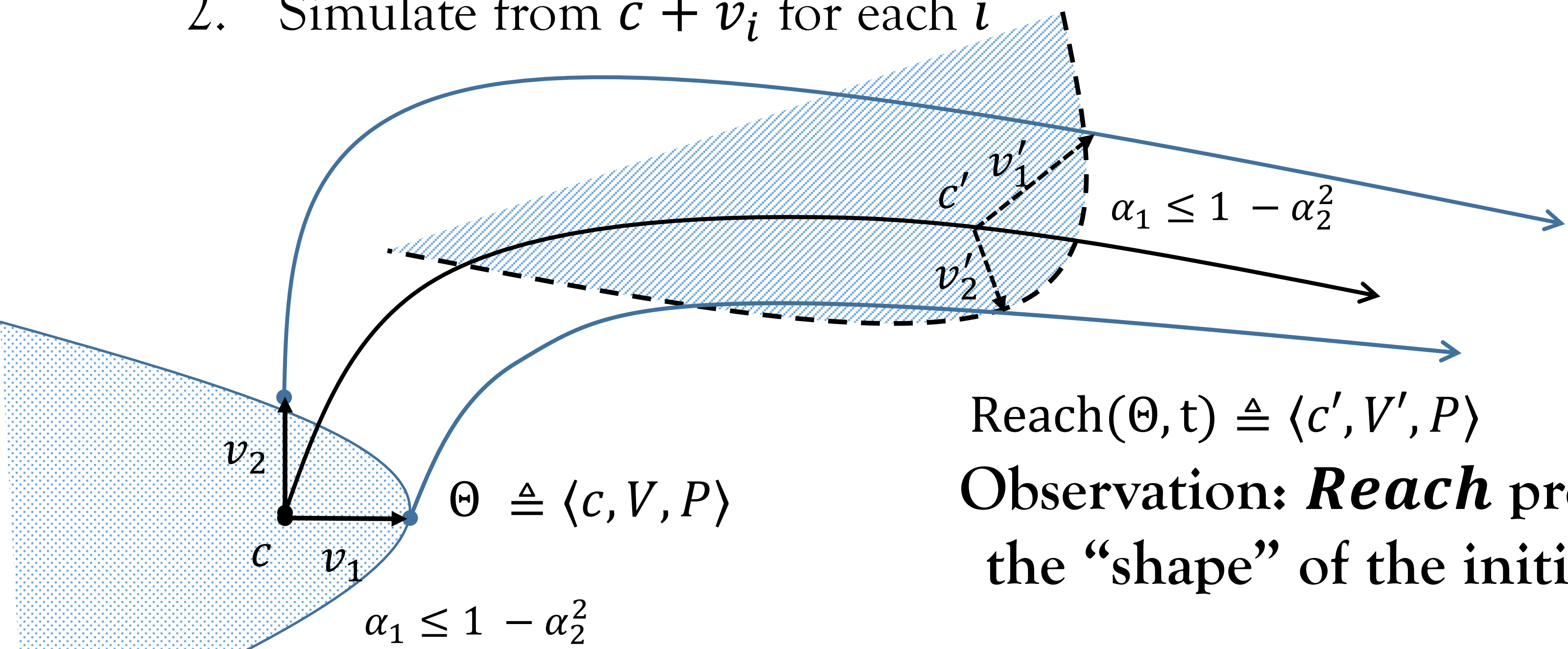
Reachable set at time t is given by $\langle c', V', P \rangle$ where

1. c' is the simulation corresponding to c
2. $v_i{}'$ is the difference of simulations from $c + v_i$ and from c

Reachable Set Computation Using Simulations For Generalized Stars

Given $\Theta \triangleq \langle c, V, P \rangle$ to compute reachable set

1. Simulate from c
2. Simulate from $c + v_i$ for each i



$\text{Reach}(\Theta, t) \triangleq \langle c', V', P \rangle$

Observation: *Reach* preserves the "shape" of the initial set.

Reachable set at time t is given by $\langle c', V', P \rangle$ where

1. c' is the simulation corresponding to c

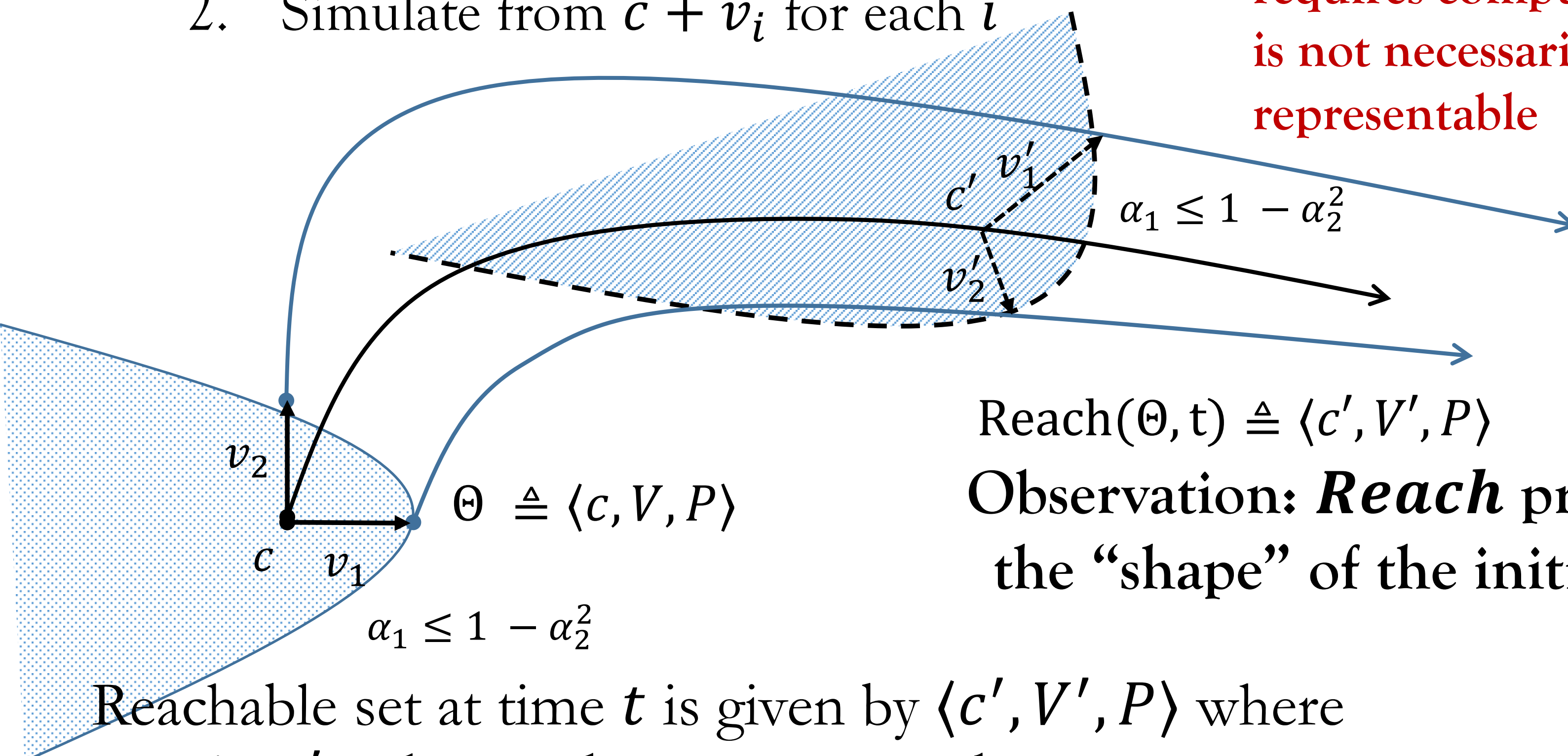
2. v_i' is the difference of simulations from $c + v_i$ and from c

Reachable Set Computation Using Simulations For Generalized Stars

Given $\Theta \triangleq \langle c, V, P \rangle$ to compute reachable set

1. Simulate from c
2. Simulate from $c + v_i$ for each i

Problem: Exact simulations requires computing e^{At} and is not necessarily finitely representable



$\text{Reach}(\Theta, \text{t}) \triangleq \langle c', V', P \rangle$

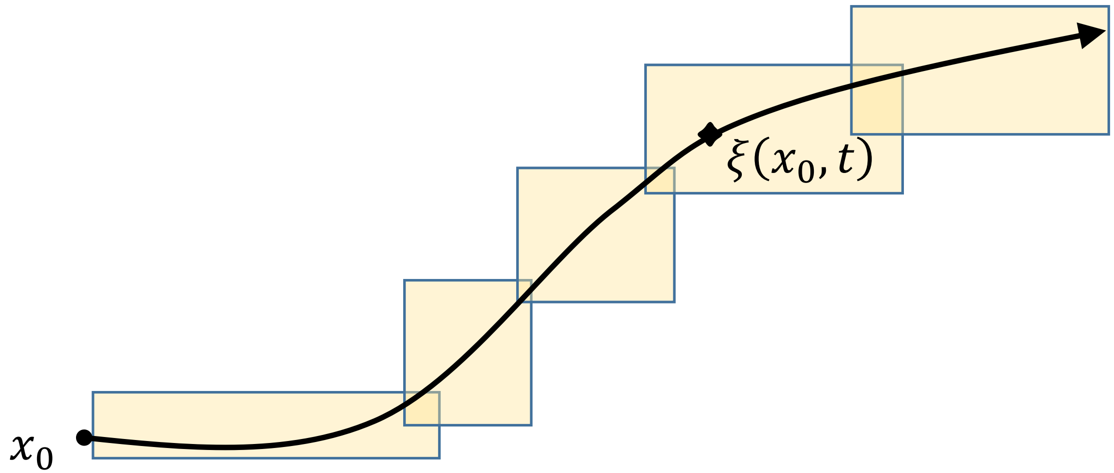
Observation: *Reach* preserves the "shape" of the initial set.

Reachable set at time t is given by $\langle c', V', P \rangle$ where

1. c' is the simulation corresponding to c

2. v_i' is the difference of simulations from $c + v_i$ and from c

Validated Simulations

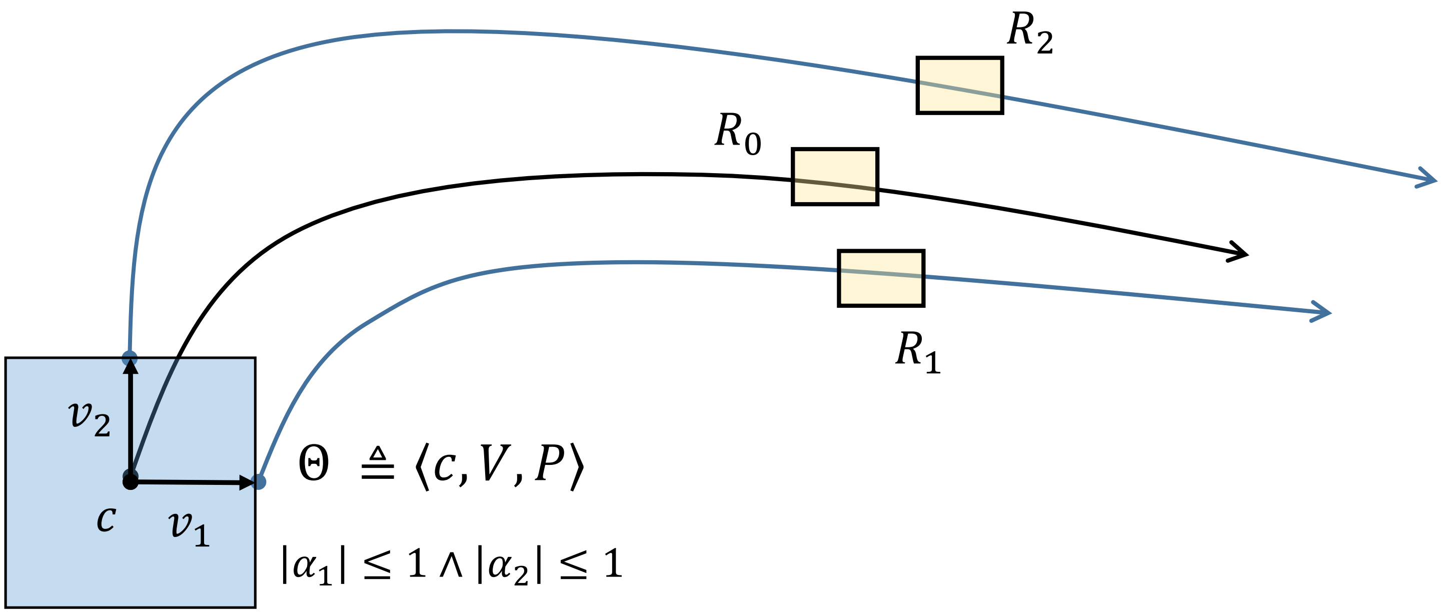


$valSim(x_0, t)$ returns sequence of regions such that
$\xi(x_0, t) \in R_l$ when $t \in [t_l, t_{l+1}]$

$diameter(R_l) \rightarrow 0$ as $|\mathrm{t}_{l+1} - t_l| \rightarrow 0$

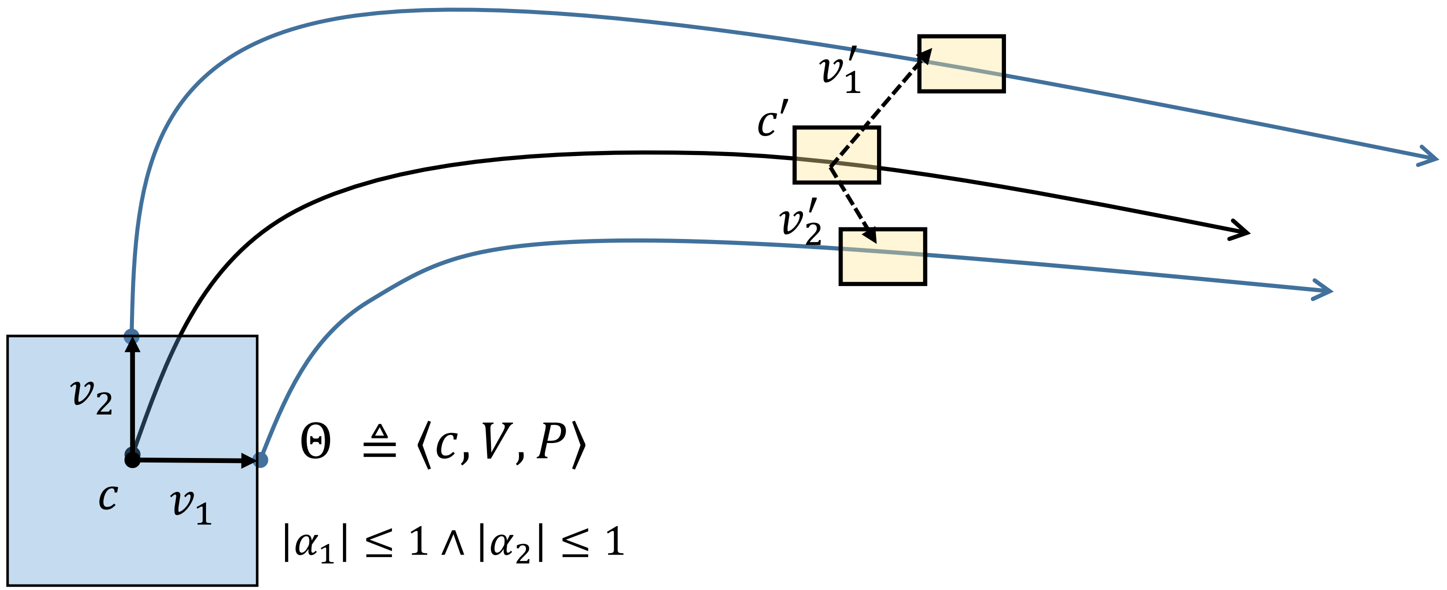
Over- and Under-Approximations Using Validated Simulations

- Problem – exact value of c, v_1', and v_2' is not known!



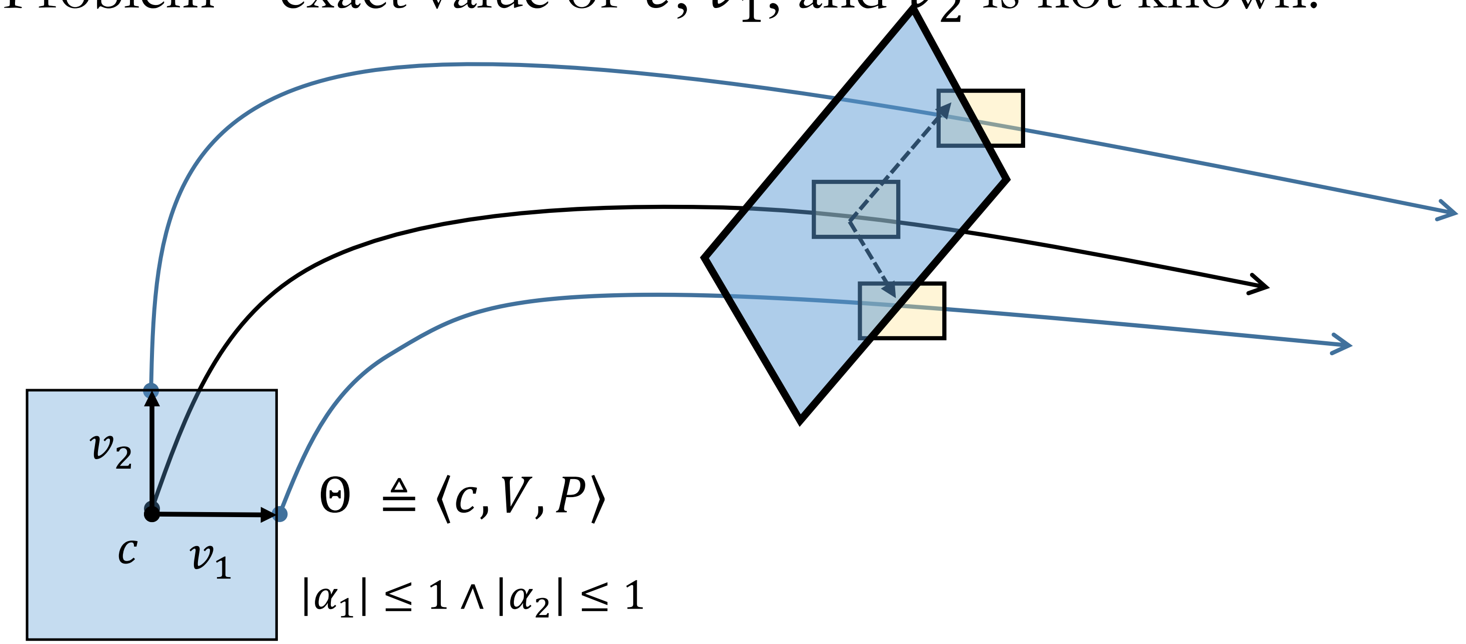
Over- and Under-Approximations Using Validated Simulations

- Problem – exact value of c, v_1', and v_2' is not known!



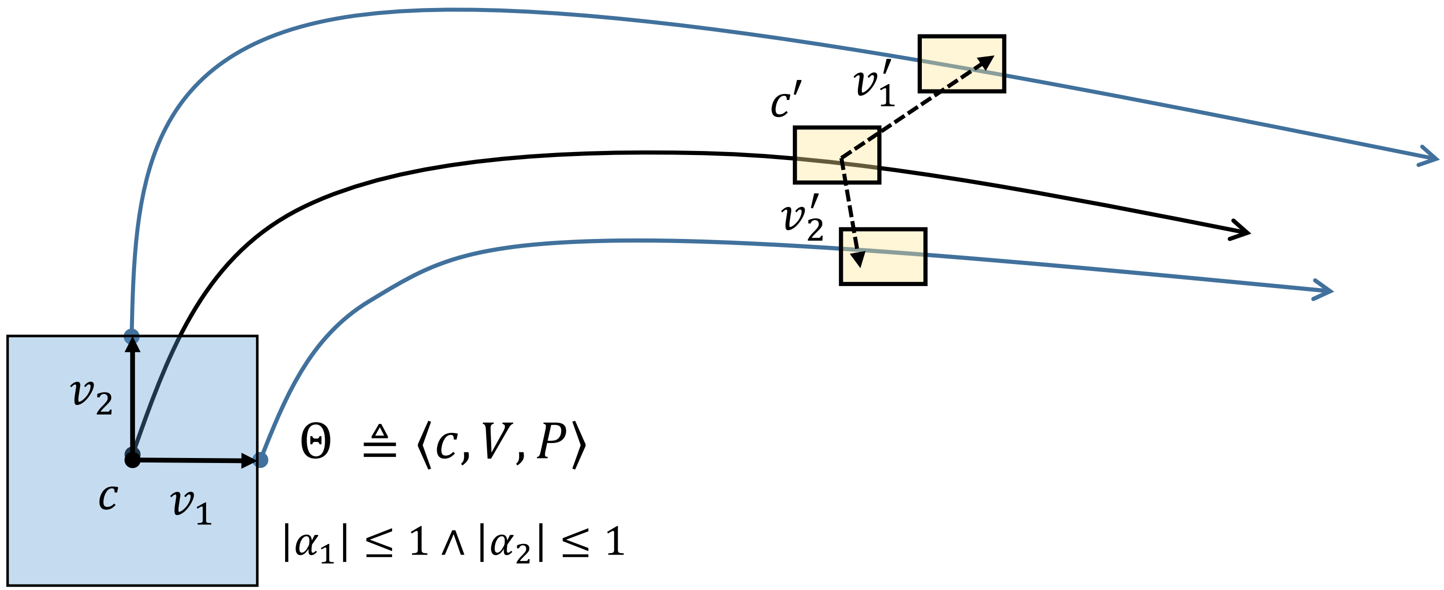
Over- and Under-Approximations Using Validated Simulations

- Problem – exact value of c, v_1', and v_2' is not known!



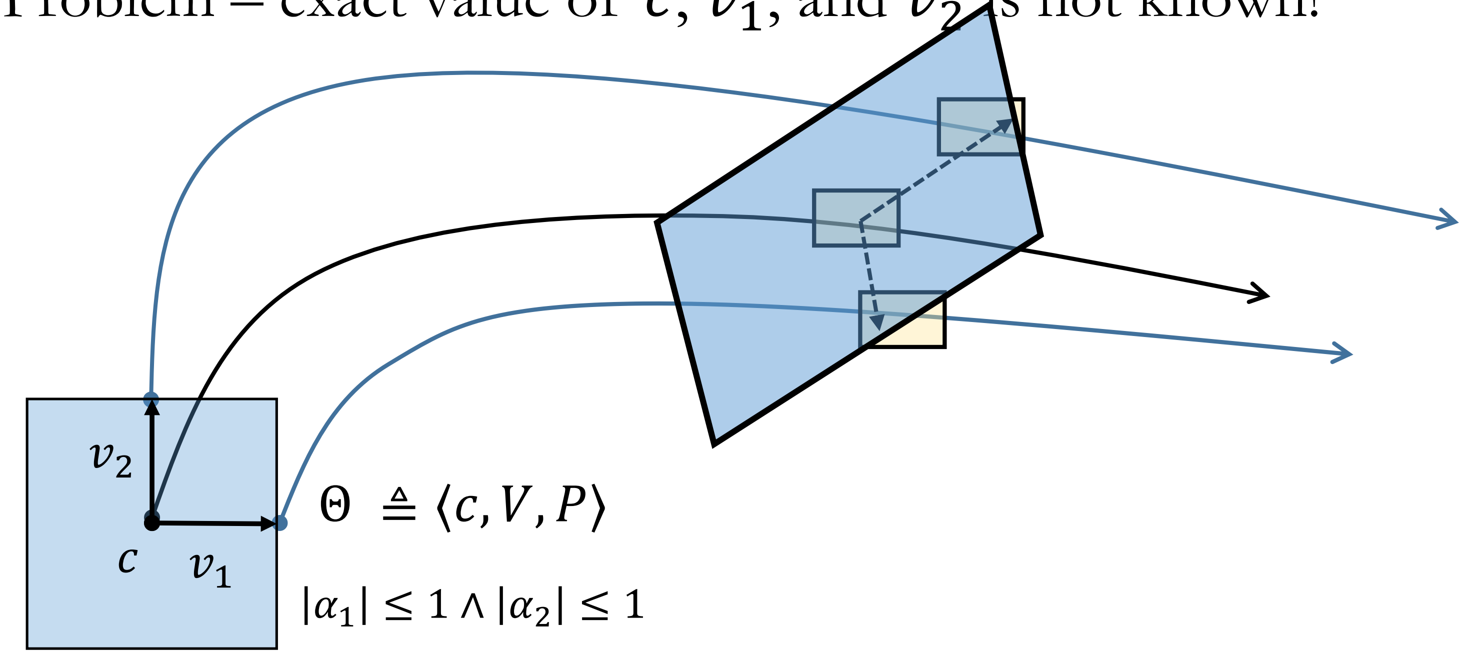
Over- and Under-Approximations Using Validated Simulations

- Problem – exact value of c, v_1', and v_2' is not known!



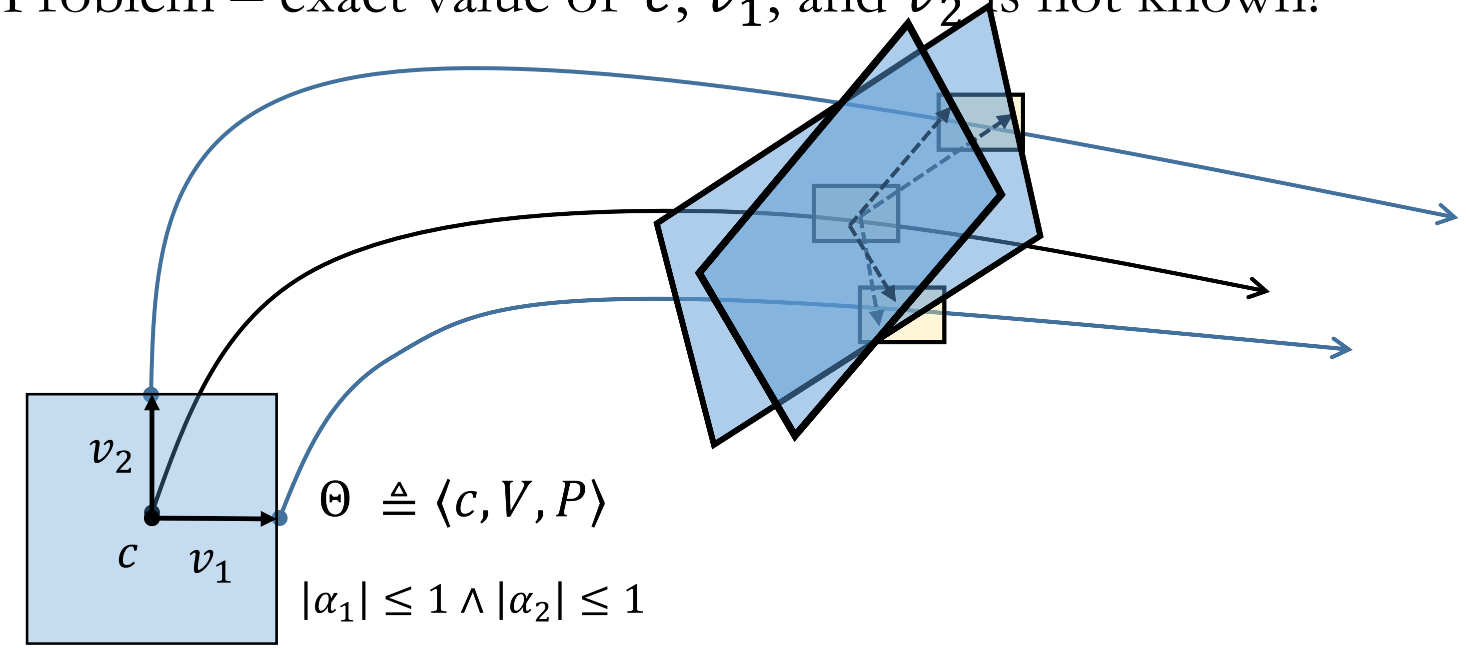
Over- and Under-Approximations Using Validated Simulations

- Problem – exact value of c, v_1', and v_2' is not known!



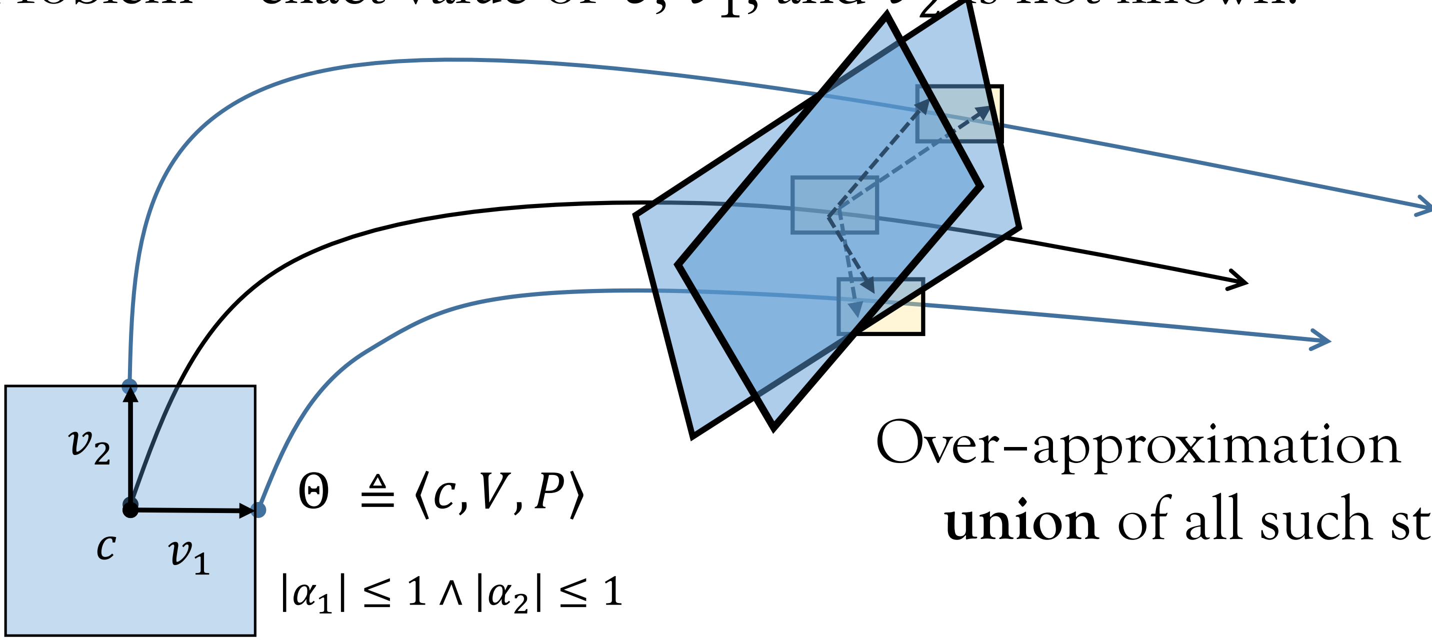
Over- and Under-Approximations Using Validated Simulations

- Problem – exact value of c, v_1', and v_2' is not known!



Over- and Under-Approximations Using Validated Simulations

- Problem – exact value of c, v_1', and v_2' is not known!

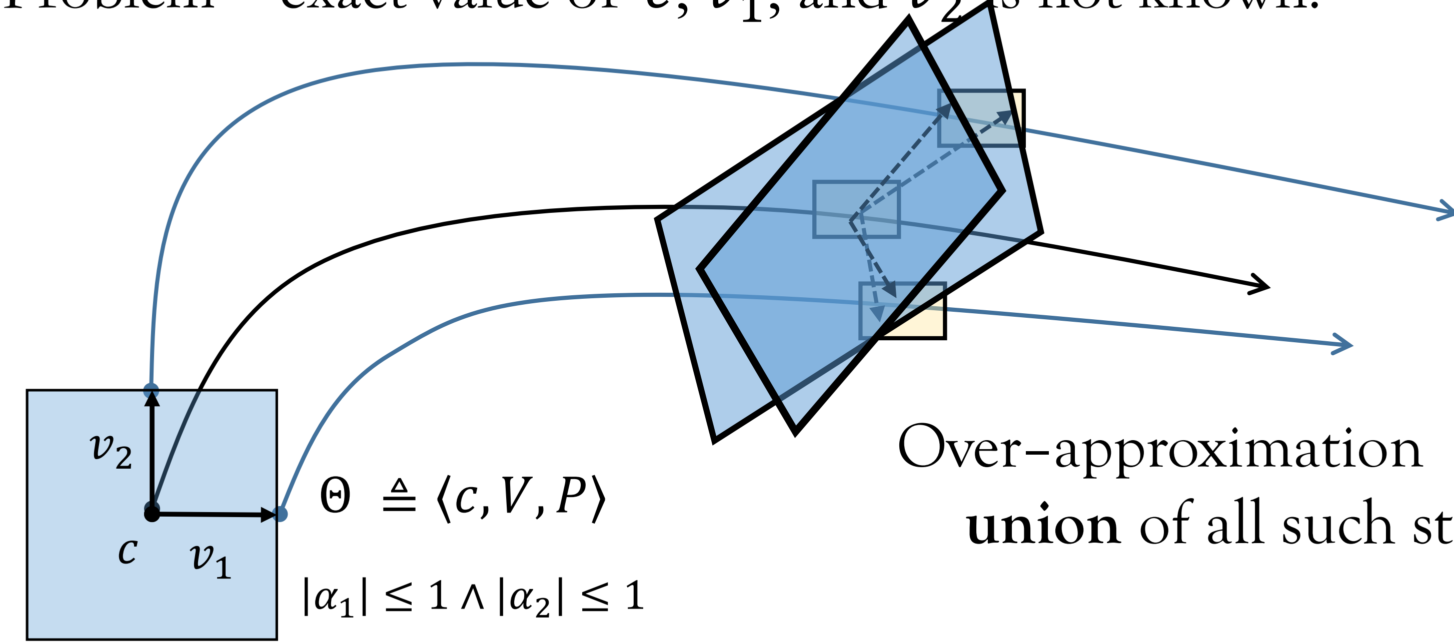


Over-approximation is the **union** of all such stars

Under-approximation is the **intersection** of all such stars

Over- and Under-Approximations Using Validated Simulations

- Problem – exact value of c, v_1', and v_2' is not known!



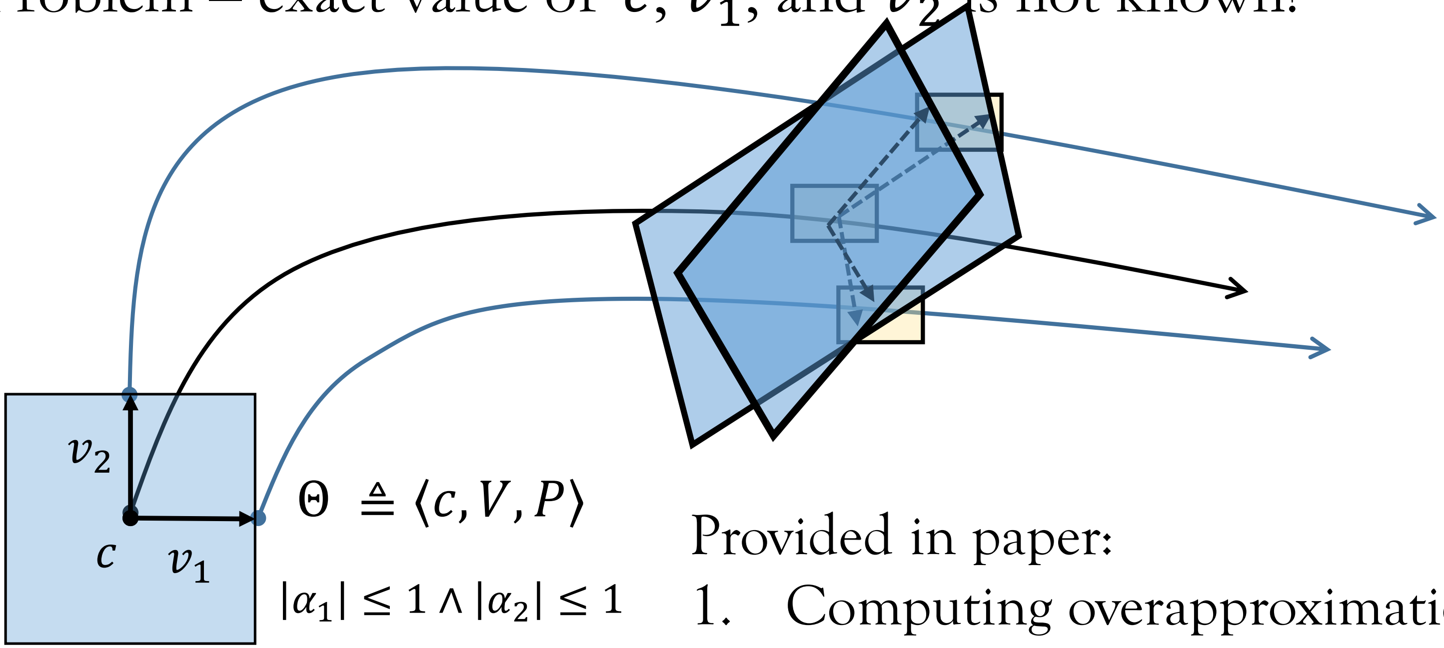
Over–approximation is the **union** of all such stars

Under–approximation is the **intersection** of all such stars

$$OA = \{x \mid \exists c, \exists v_1, \exists v_2\ \exists \bar{\alpha}, x = c + \alpha_1 v_1 + \alpha_2 v_2\}$$
$$UA = \{x \mid \forall c, \forall v_1, \forall v_2\ \exists \bar{\alpha}, x = c + \alpha_1 v_1 + \alpha_2 v_2\}$$

Over- and Under-Approximations Using Validated Simulations

- Problem – exact value of c, v_1', and v_2' is not known!



Provided in paper:
1. Computing overapproximation
2. Checking safety violation

without using QE for bounded initial sets

Experimental Results - I

Comparison with SpaceEx on Linear Systems

Benchmark	Vars.	TH	Sim.Time.	Verif.Time.	C2E2	SpaceEx	Result
Insulin	8	10	0.157 s	0.049 s	**0.20 s**	8.07 s	Safe
Inslin	8	10	0.166 s	0.034 s	**0.2 s**	7.89 s	Unsafe
Platoon	10	25	0.337 s	0.019 s	**0.356 s**	TO	Safe
Platoon	10	25	0.323 s	0.019 s	**0.342 s**	TO	Unsafe
Tank-10	10	20	0.745 s	0.206 s	**0.951 s**	4.886 s	Safe
Tank-10	10	20	0.721 s	0.19 s	**0.911 s**	4.992 s	Unsafe
Tank-15	15	20	1.325 s	0.363 s	**1.688 s**	8.176 s	Safe
Tank-18	18	20	1.705 s	0.569 s	**2.274 s**	10.466 s	Safe
Helicopter	28	20	3.192 s	1.634 s	**4.826 s**	2m 1.66 s	Safe

Experimental Results - II

Comparison with Flow* for Linear Time Varying Systems

Benchmark	Vars	C2E2	Flow*
Tank-TV	2	0.132 s	1.56 s
Tank-TV	4	0.198 s	4.28 s
Tank-TV	6	0.287 s	9.41 s
Tank-TV	8	0.356 s	18.73 s
Tank-TV	10	0.484 s	33.67 s
LTV	5	0.24 s	7.51 s
LTV	7	0.31 s	12.09 s
LTV	9	0.4 s	18.18 s

Experimental Results - III

Verification of Non-convex and Unbounded Initial Sets

Benchmark	Dim.	TH.	Init. Set.	Res.	Time
ACC	3	2	Non-Convex	Safe	**2.185**
ACC	3	2	Unbounded	Safe	**1.774**
ACC	3	2	Non-Convex	Unsafe	**1.11**
ACC	3	2	Unbounded	Unsafe	**1.01**
Tank	5	1	Non-Convex	Safe	**2.717**
Tank	5	1	Unbounded	Safe	**2.145**
Tank	5	1	Non-Convex	Unsafe	**1.722**
Tank	5	1	Unbounded	Unsafe	**1.519**

Conclusions

- New simulation based verification for linear systems
    1. For n–dimensional system, $n + 1$ simulations suffice.
    2. Works for both time invariant and time variant systems.
    3. Works for non-convex and unbounded systems
    4. Can compute over- and under-approximation.
    5. Reuse the simulations for different initial sets.

Thank You