What's to Come is Still Unsure *

Synthesizing Controllers Resilient to Delayed Interaction

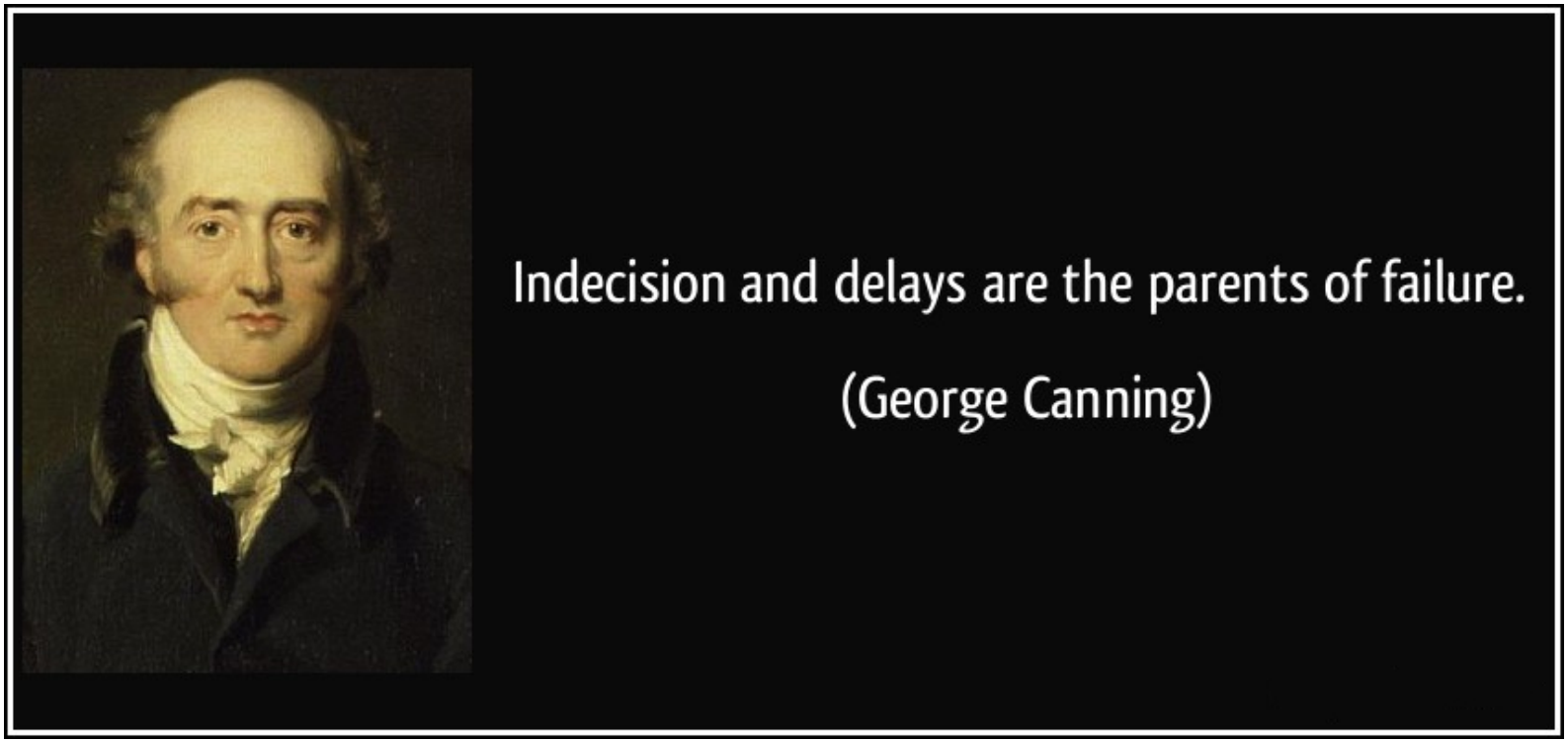
Mingshuai Chen

—Joint work with Martin Fränzle, Yangjia Li, Peter N. Mosaad, Naijun Zhan—

MISSION · INVAP · February 2022

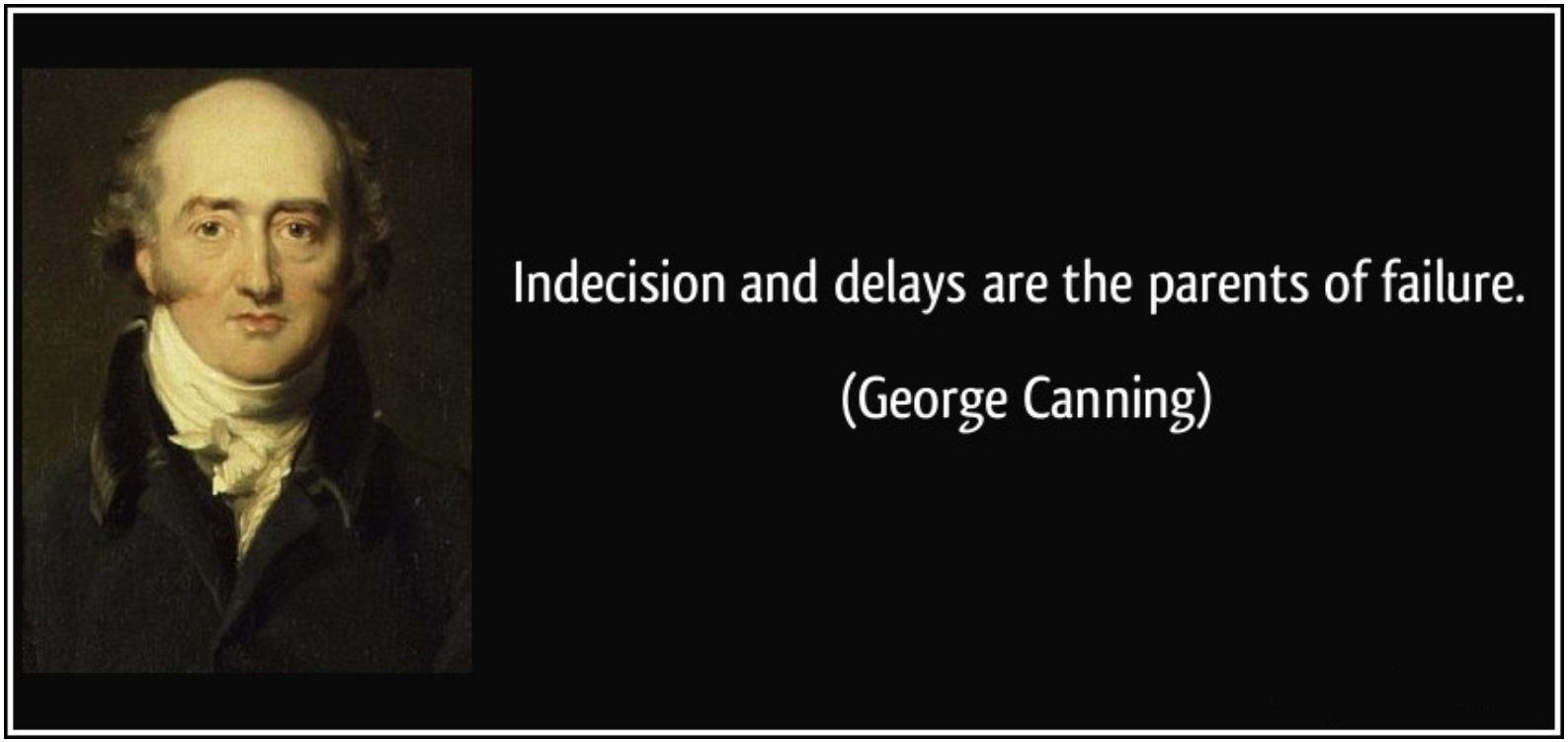
* William Shakespeare, Twelfth Night/What You Will, Act 2, Scene 3.

A Pearl of Wisdom



©izQuotes

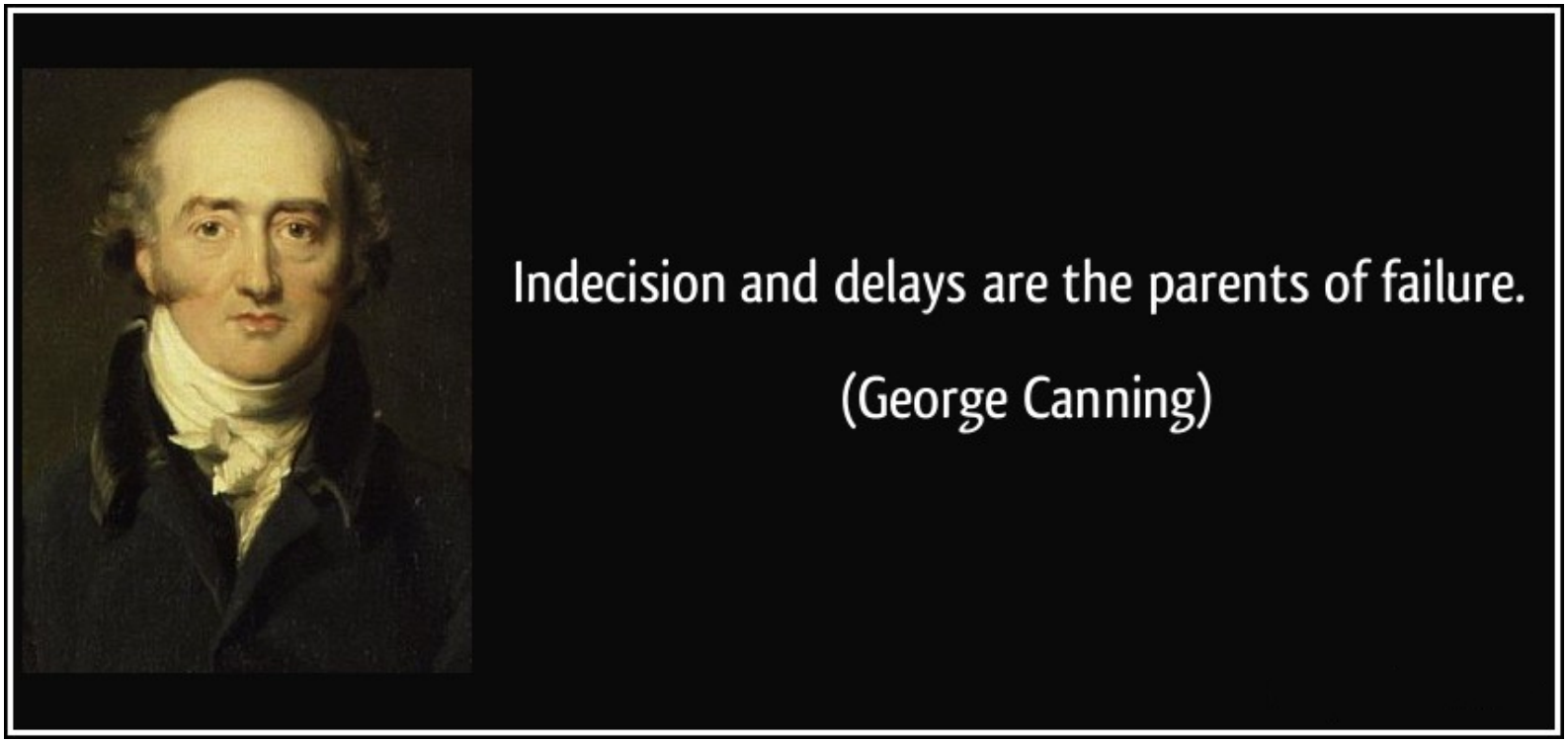
A Pearl of Wisdom



©izQuotes

- Only relevant to ordinary people's life?

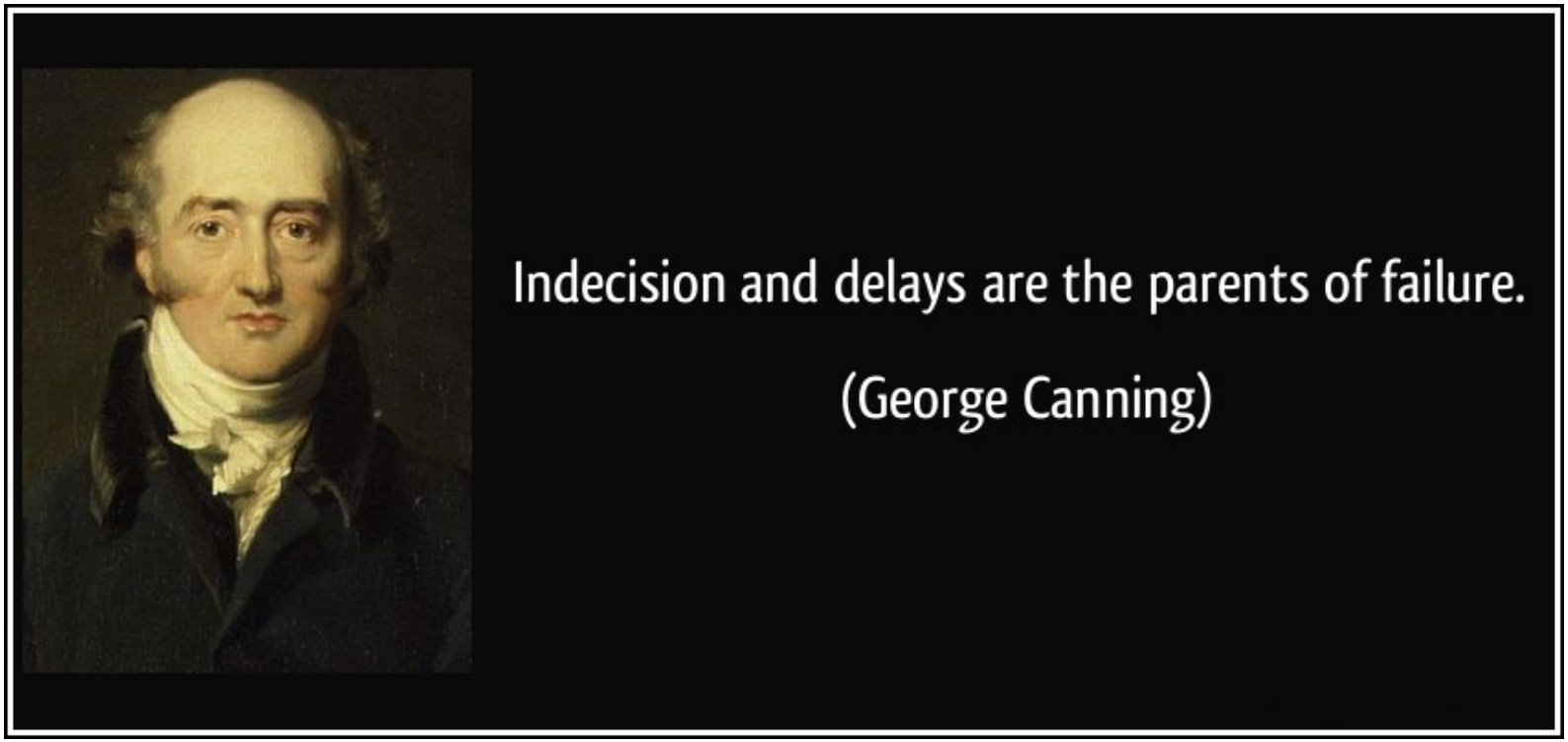
A Pearl of Wisdom



©izQuotes

- Only relevant to ordinary people's life?
- Or to scientists, in particular **comp. sci.** and **control folks**, too?

A Pearl of Wisdom



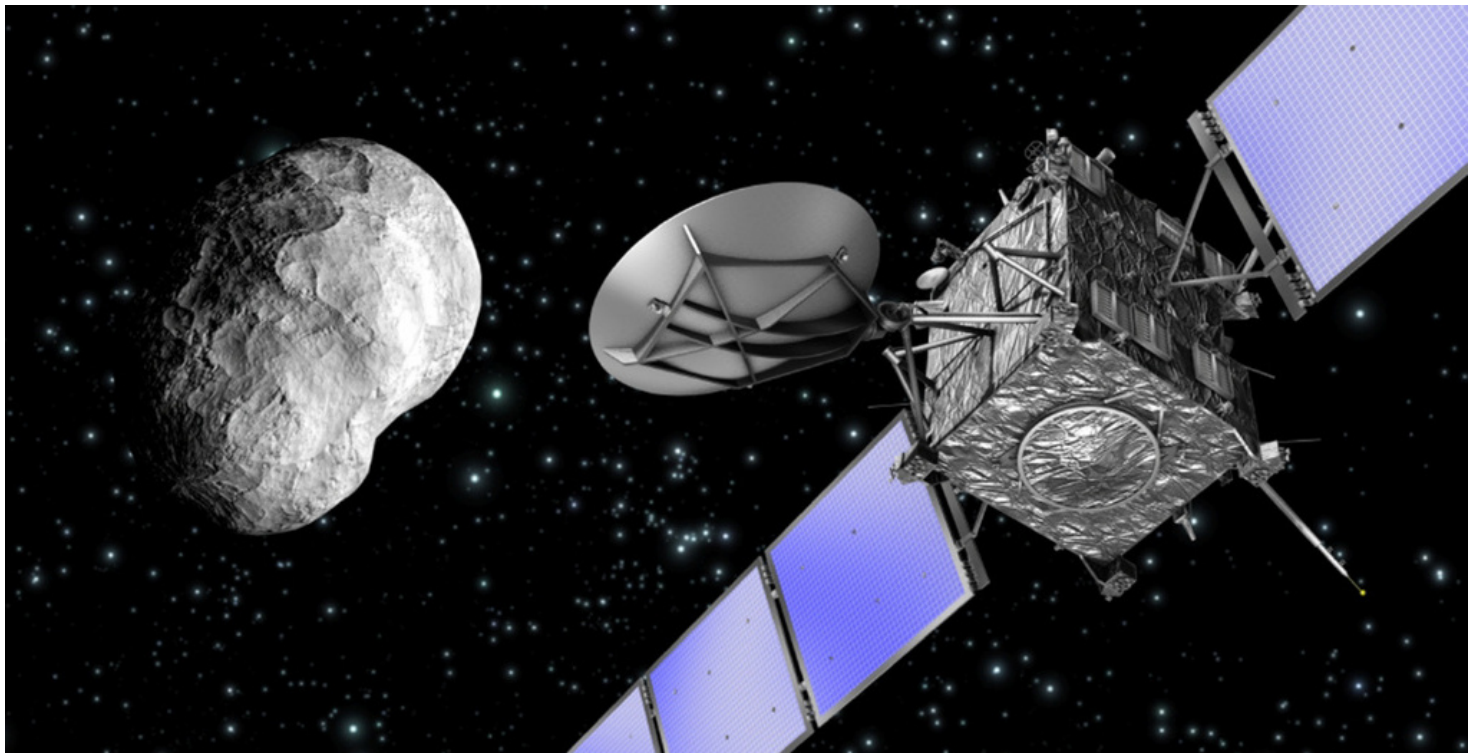
©izQuotes

- Only relevant to ordinary people's life?
- Or to scientists, in particular **comp. sci.** and **control folks**, too?

Remember that Canning briefly **controlled** Great Britain!

Staying Safe

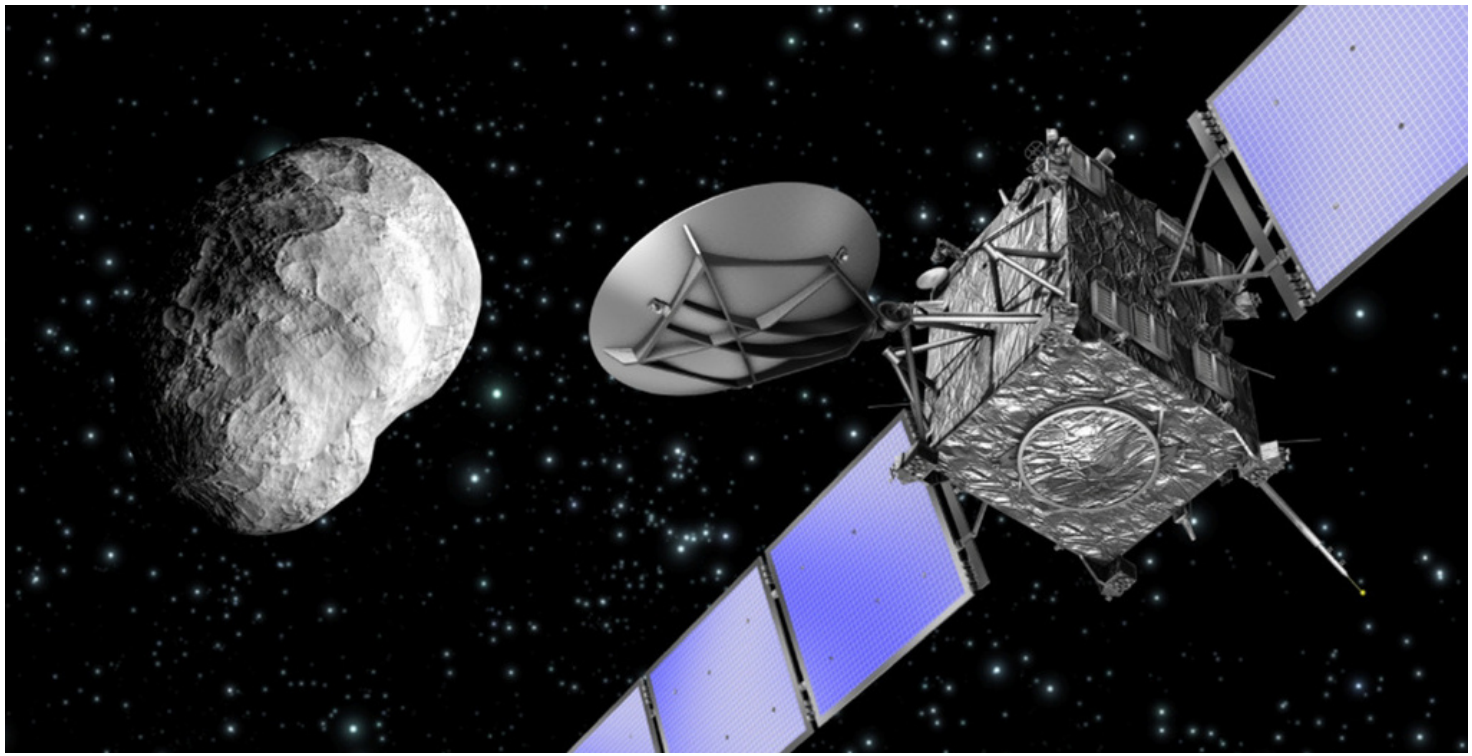
When Observation & Actuation Suffer from Serious Delays



©ESA

Staying Safe

When Observation & Actuation Suffer from Serious Delays

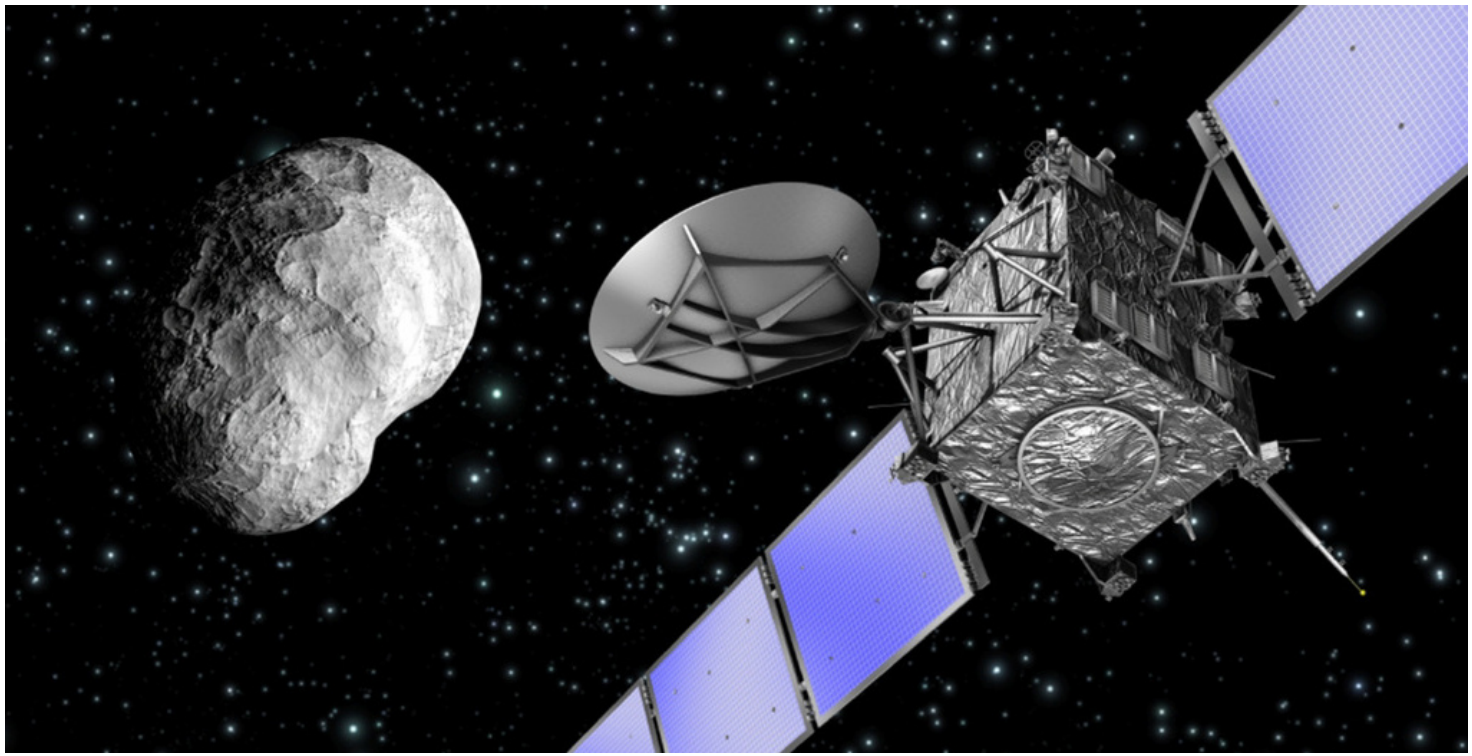


©ESA

- You could move slowly. (Well, can you?)

Staying Safe

When Observation & Actuation Suffer from Serious Delays

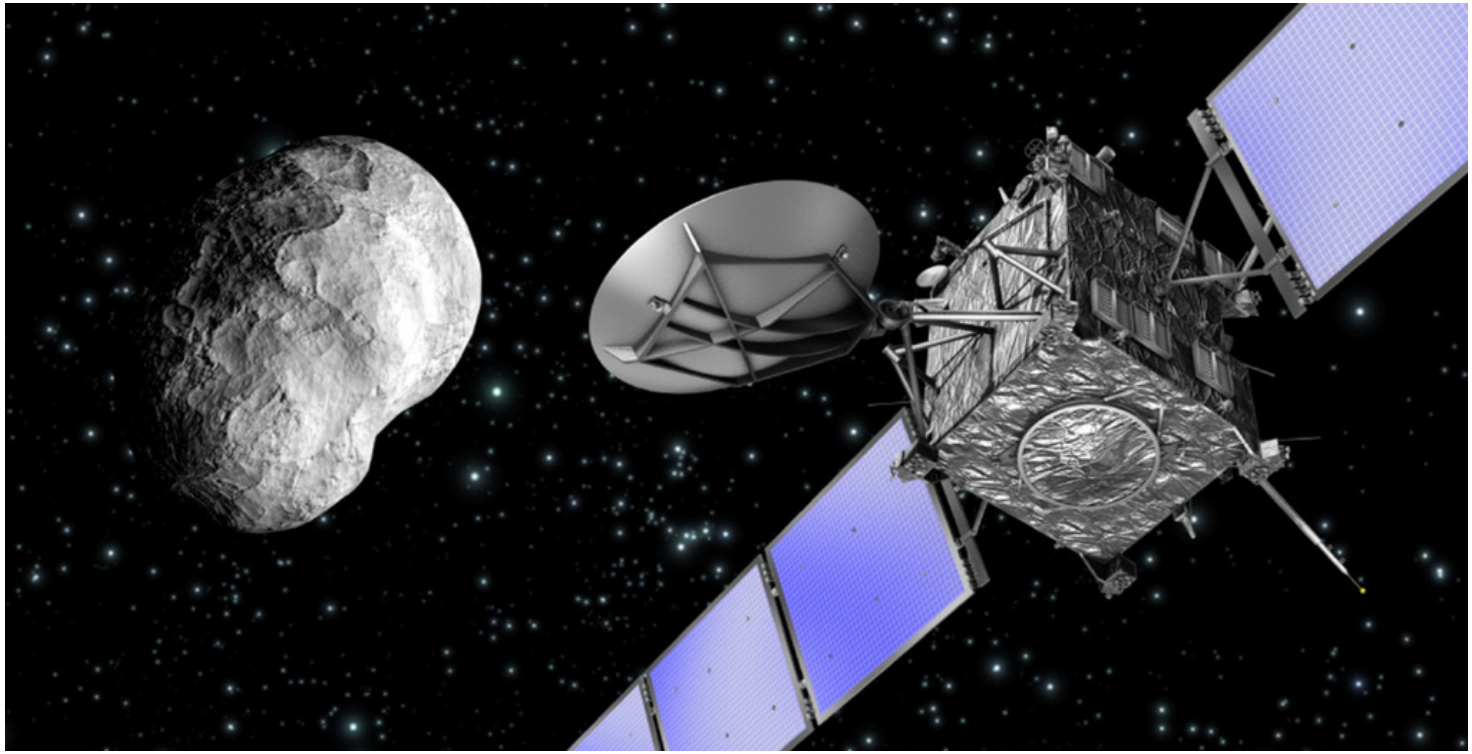


©ESA

- You could move slowly. (Well, can you?)
- You could trust autonomy.

Staying Safe

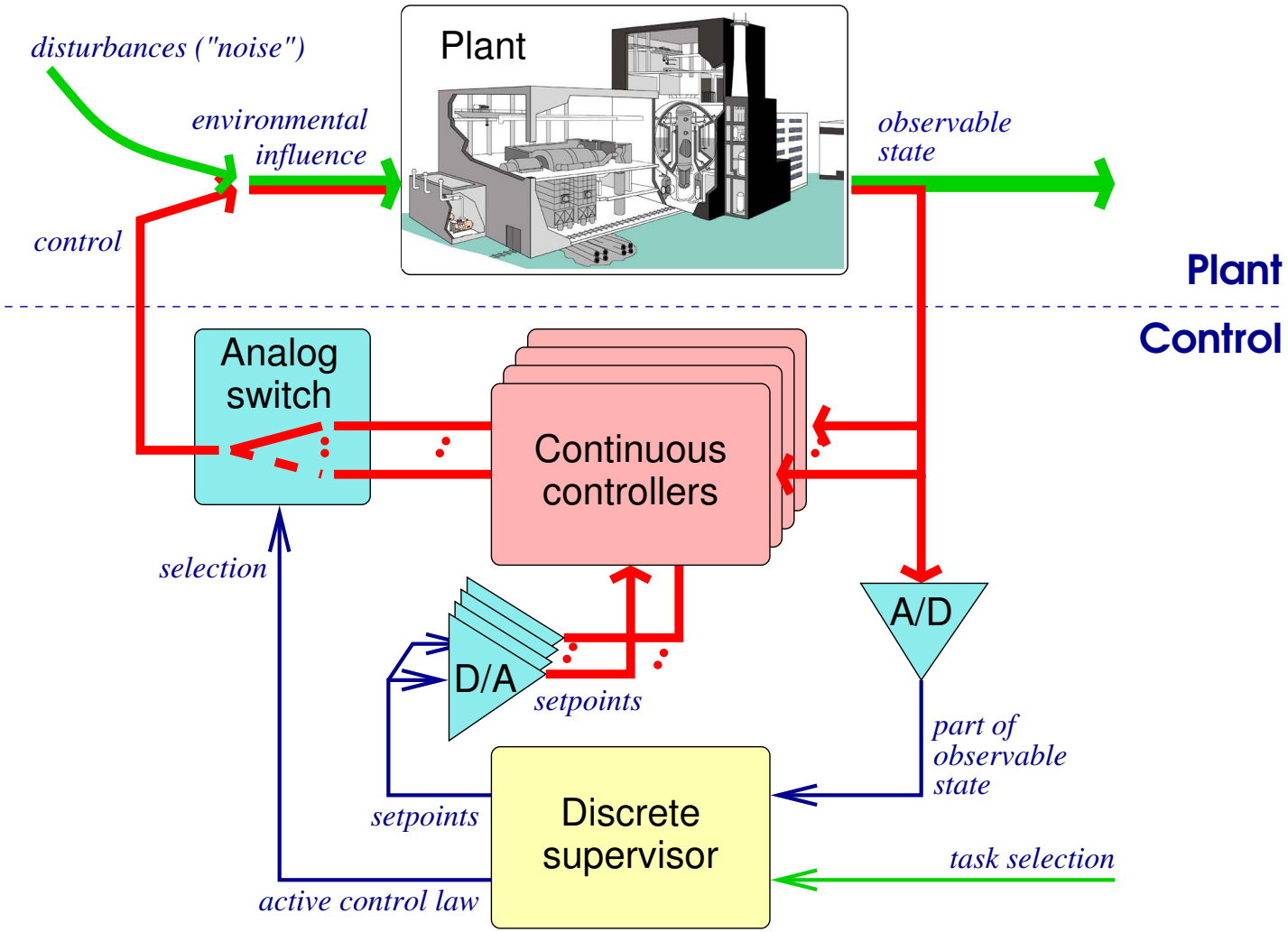
When Observation & Actuation Suffer from Serious Delays



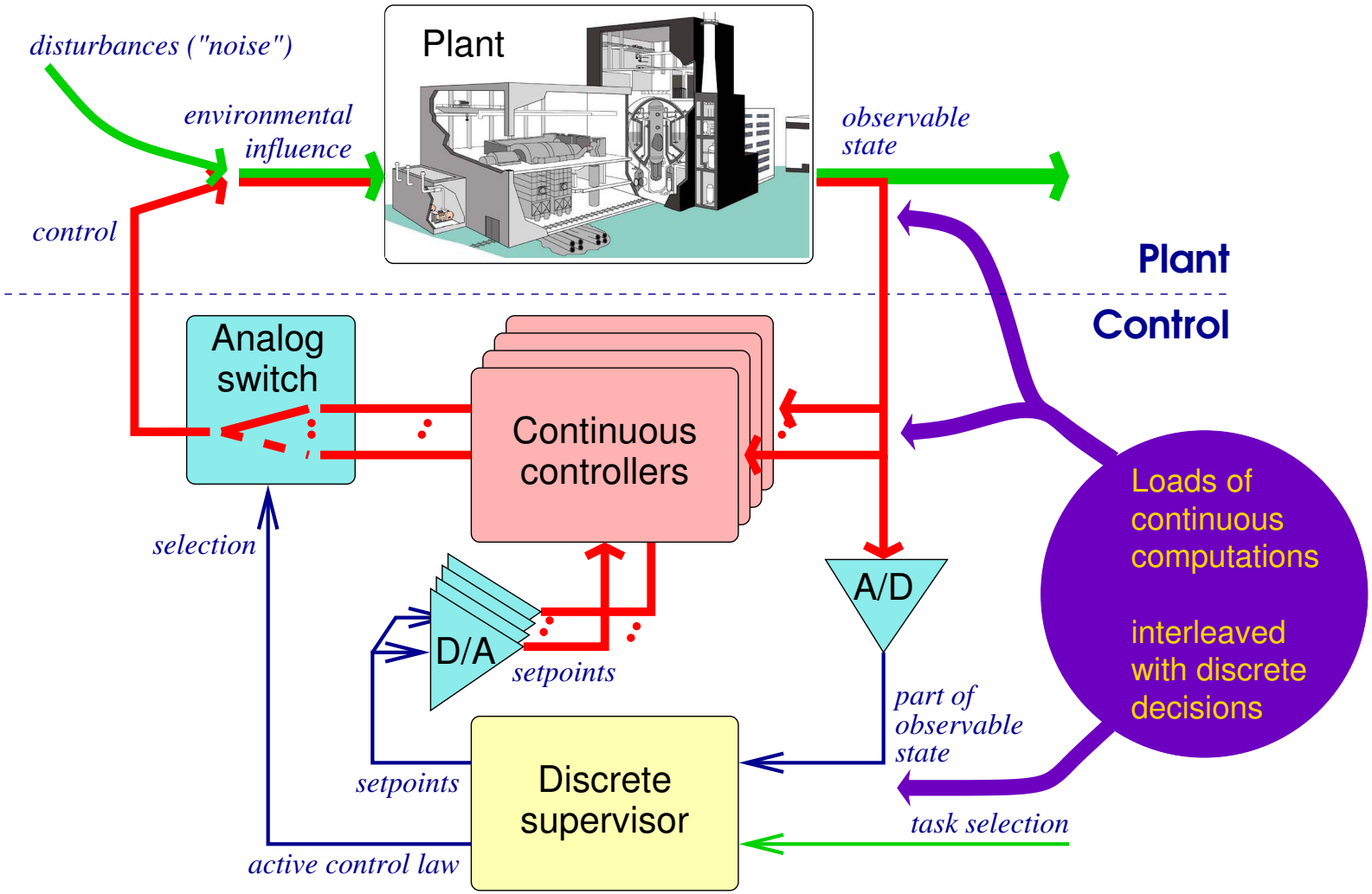
©ESA

- You could move slowly. (Well, can you?)
- You could trust autonomy.
- Or you have to **anticipate** and **issue actions early**.

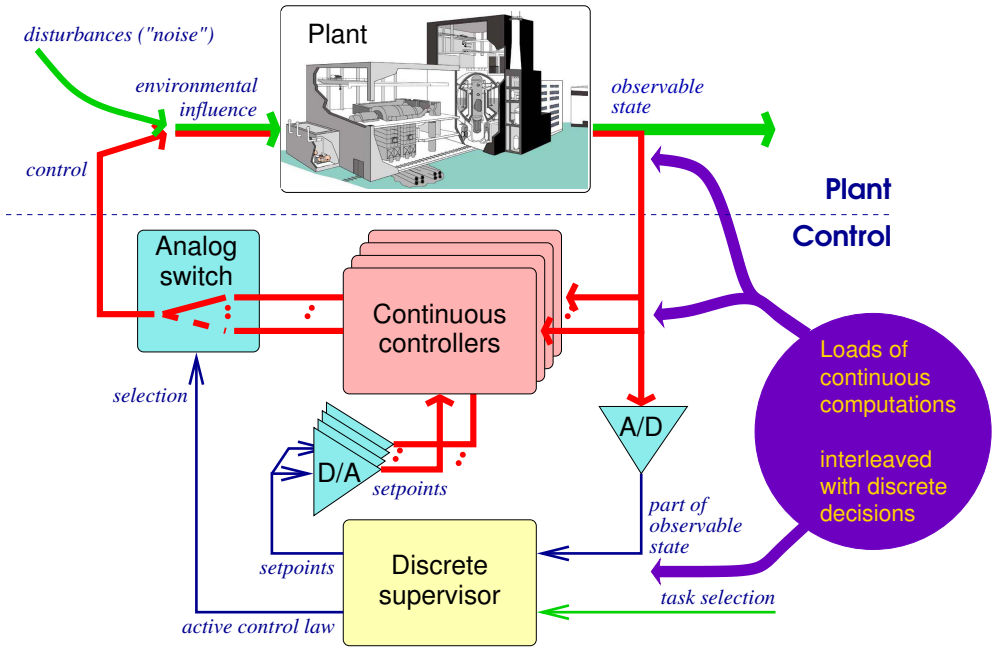
Interaction btw. a Controller and Its Environment



Interaction btw. a Controller and Its Environment



Interaction btw. a Controller and Its Environment

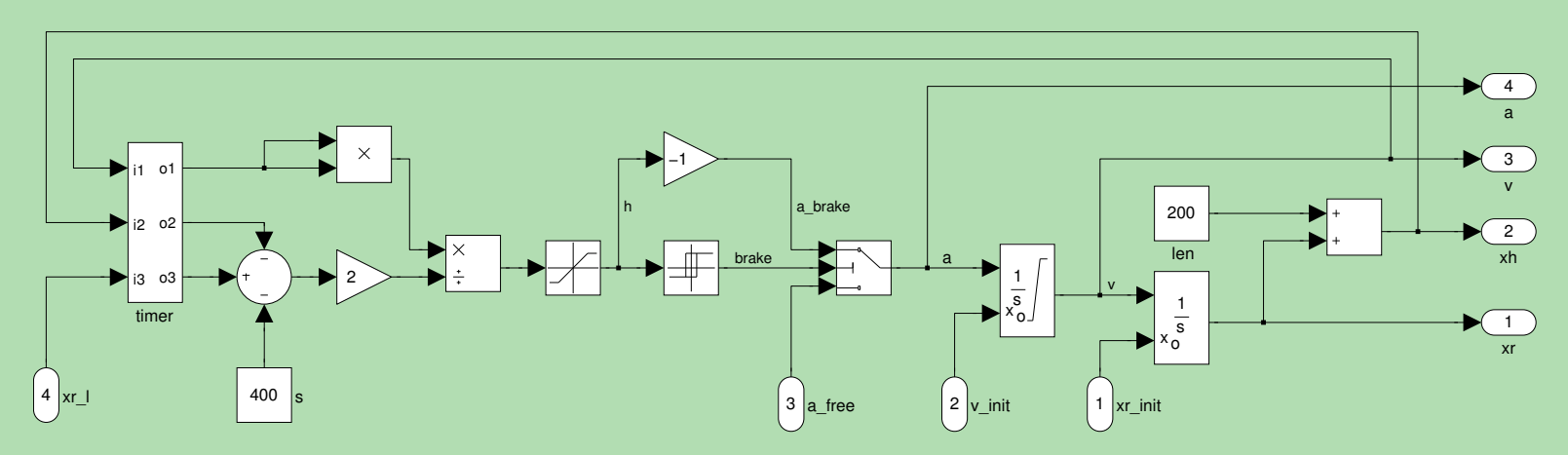


Crucial question : How do the controller and the plant interact?

Traditional answer : Coupling assumed to be (or at least modeled as) **delay-free** :

- **mode dynamics** is covered by the conjunction of individual ODEs;
- **switching btw. modes** is an immediate reaction to environmental conditions.

Instantaneous Coupling

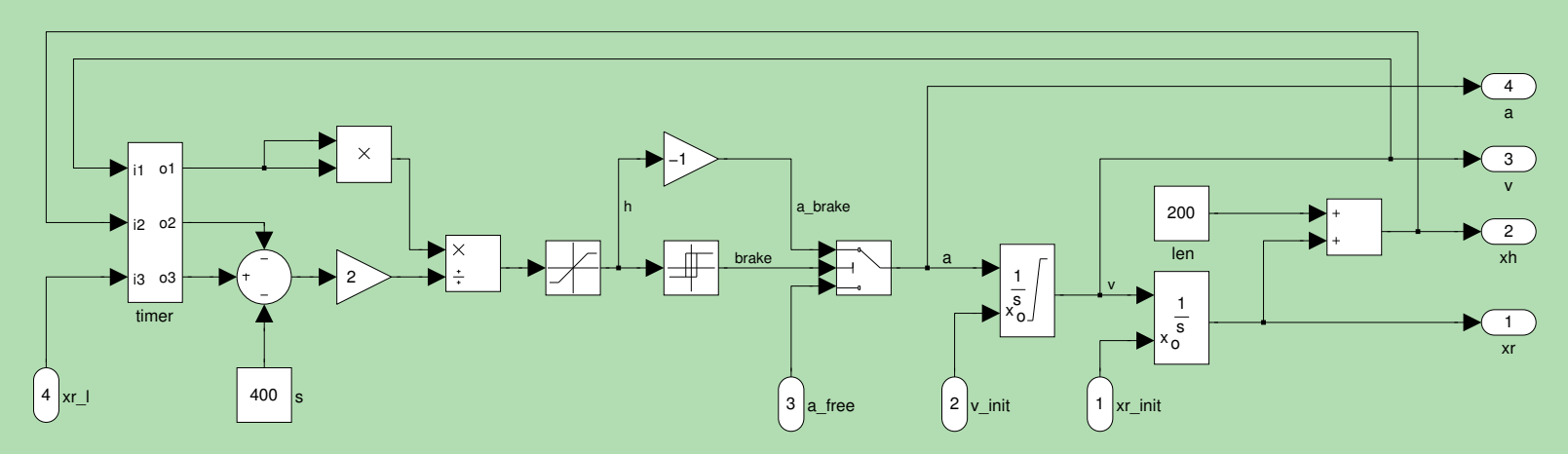


©ETCS-3

Following the tradition, the above (rather typical) Simulink model assumes

- delay-free coupling between all components;
- instantaneous feed-through within all functional blocks.

Instantaneous Coupling



©ETCS-3

Following the tradition, the above (rather typical) Simulink model assumes

- delay-free coupling between all components;
- instantaneous feed-through within all functional blocks.

Central questions :

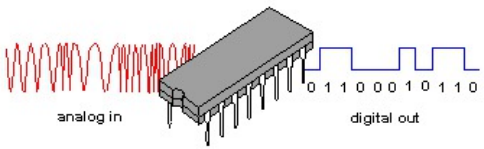
1. Is this realistic?
2. If not, does it have observable effects on control performance?
3. May those effects be detrimental or even harmful?

Q1 : Is Instantaneous Coupling Realistic?

Q1 : Is Instantaneous Coupling Realistic?

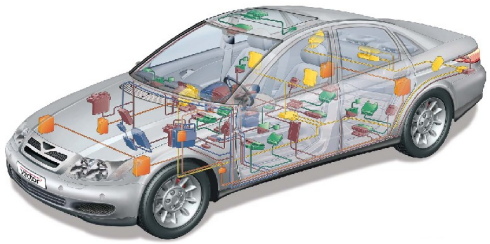
We are no better :
As soon as computer scientists enter the scene, serious delays are ahead …

Q1 : Is Instantaneous Coupling Realistic?



Digital control needs **A/D and D/A conversion**, which induces latency in signal forwarding.

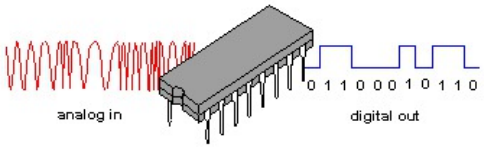
Digital **signal processing**, especially in complex sensors like CV, needs processing time, adding signal delays.



Networked control introduces communication latency into the feedback control loop.

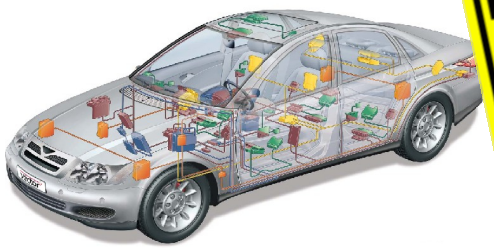
Harvesting, fusing, and forwarding data through **sensor networks** enlarge the communication latency by orders of magnitude.

Q1 : Is Instantaneous Coupling Realistic? – No.



Digital control needs **A/D and D/A conversion**, which induces latency in signal forwarding.

Digital **sig** pecially in complex sensors
e, adding signal delays.



communication latency
p.

Harvesting, fusing, and forwarding data through **sensor networks** enlarge the communication latency by orders of magnitude.

Q2 : Do Delays Have Observable Effects?

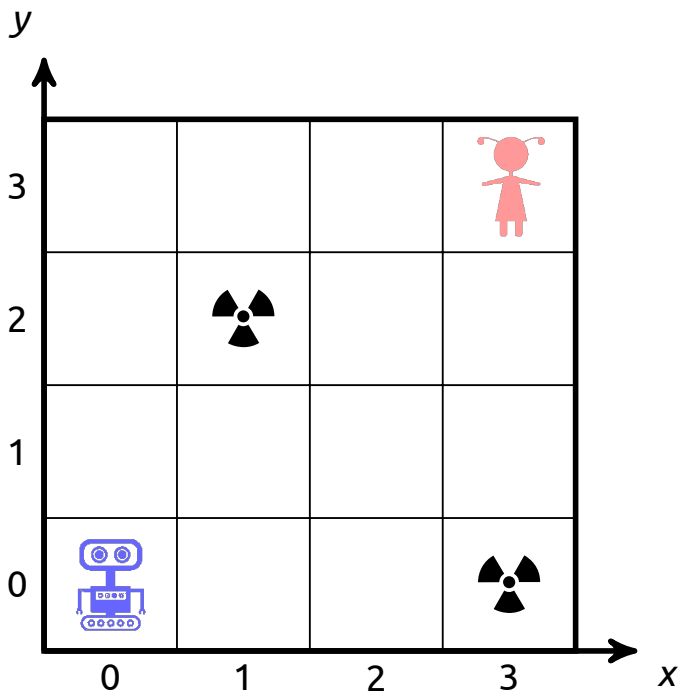


Figure – A robot escape game in a 4×4 room.

Q2 : Do Delays Have Observable Effects?

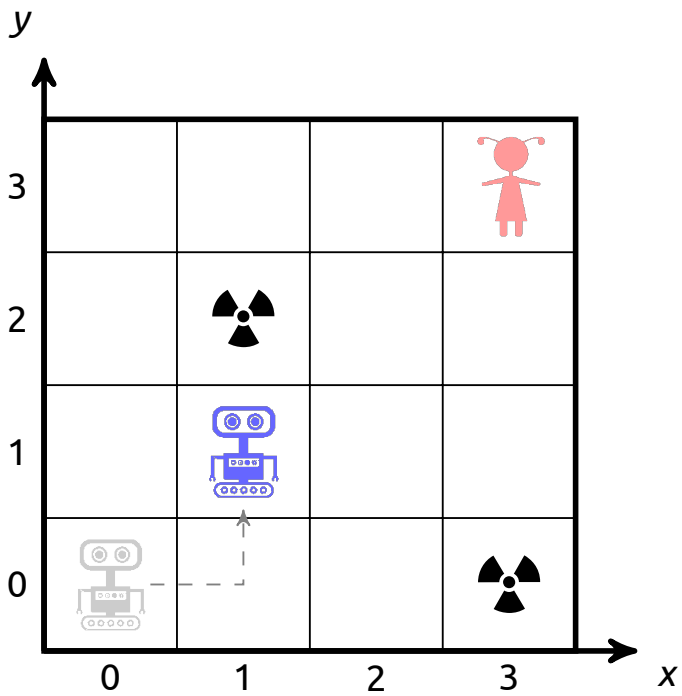


Figure – A robot escape game in a 4×4 room.

Q2 : Do Delays Have Observable Effects?

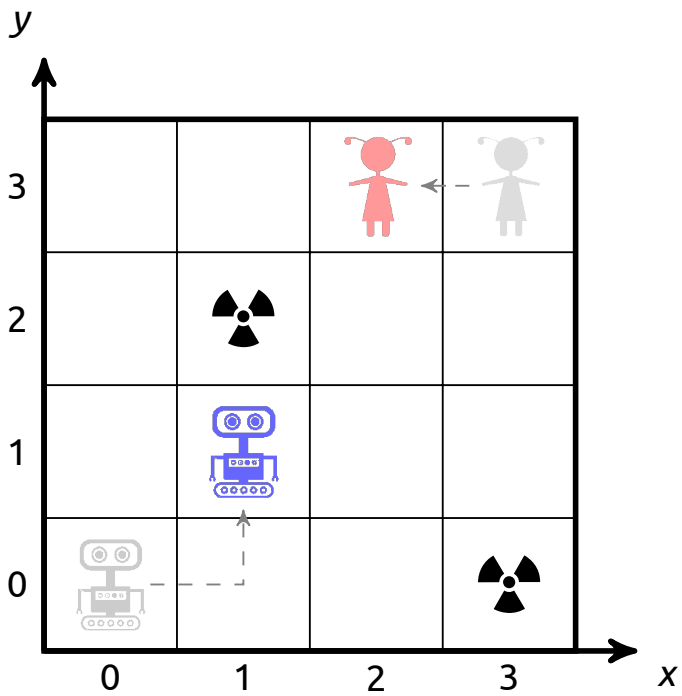


Figure – A robot escape game in a 4×4 room.

Q2 : Do Delays Have Observable Effects?

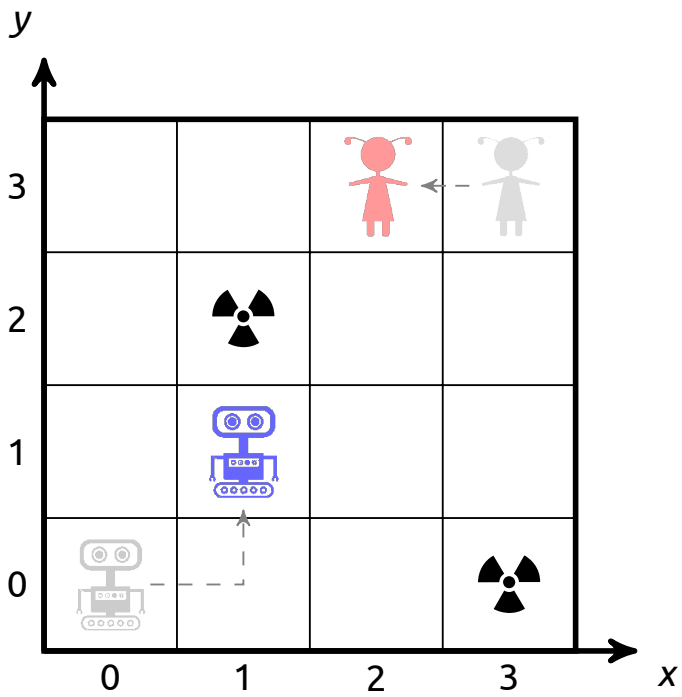


Figure – A robot escape game in a 4×4 room.

No delay :

Q2 : Do Delays Have Observable Effects?

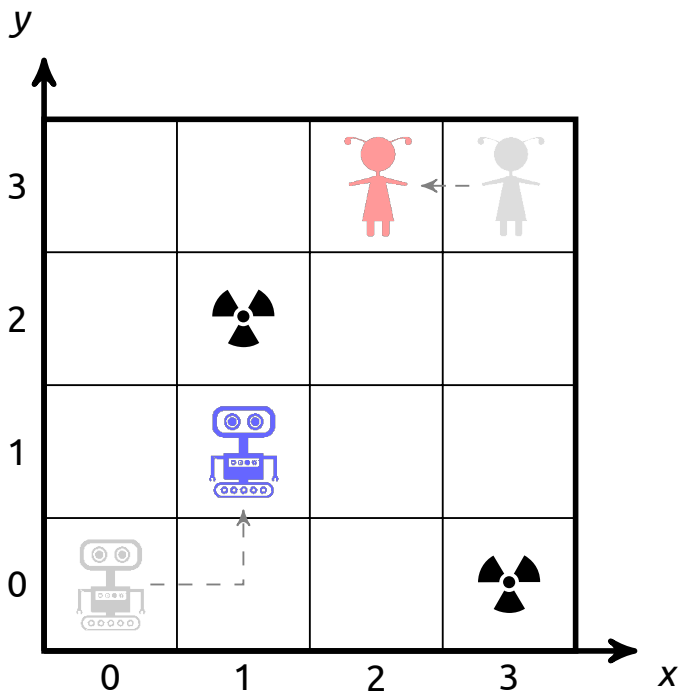


Figure – A robot escape game in a 4×4 room.

No delay :

Robot always wins by circling around the obstacle ☢ at (1,2).

Q2 : Do Delays Have Observable Effects?

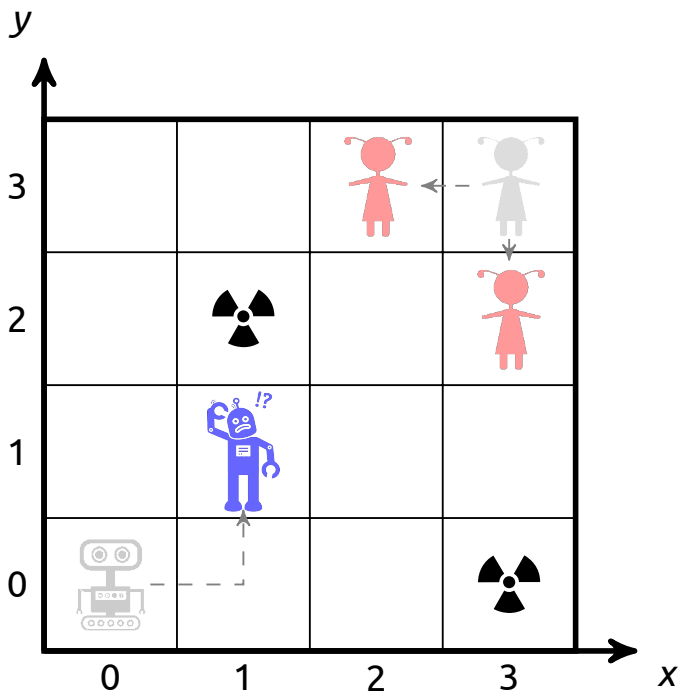


Figure – A robot escape game in a 4×4 room.

No delay :

Robot always wins by circling around the obstacle ☢ at (1,2).

1 step delay :

Q2 : Do Delays Have Observable Effects?

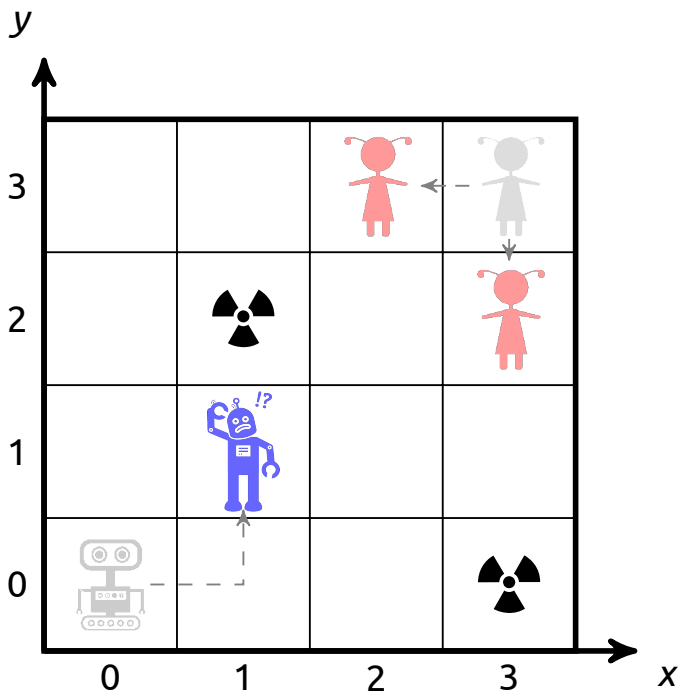


Figure – A robot escape game in a 4×4 room.

No delay :

Robot always wins by circling around the obstacle ☢ at (1,2).

1 step delay :

Robot wins by 1-step pre-decision.

Q2 : Do Delays Have Observable Effects?

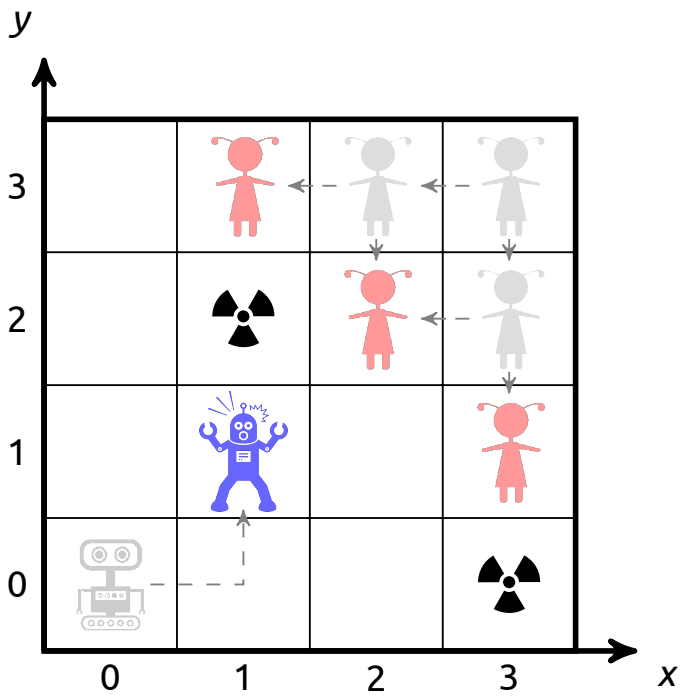


Figure – A robot escape game in a 4×4 room.

No delay :

Robot always wins by circling around the obstacle ☢ at (1,2).

1 step delay :

Robot wins by 1-step pre-decision.

2 steps delay :

Q2 : Do Delays Have Observable Effects?

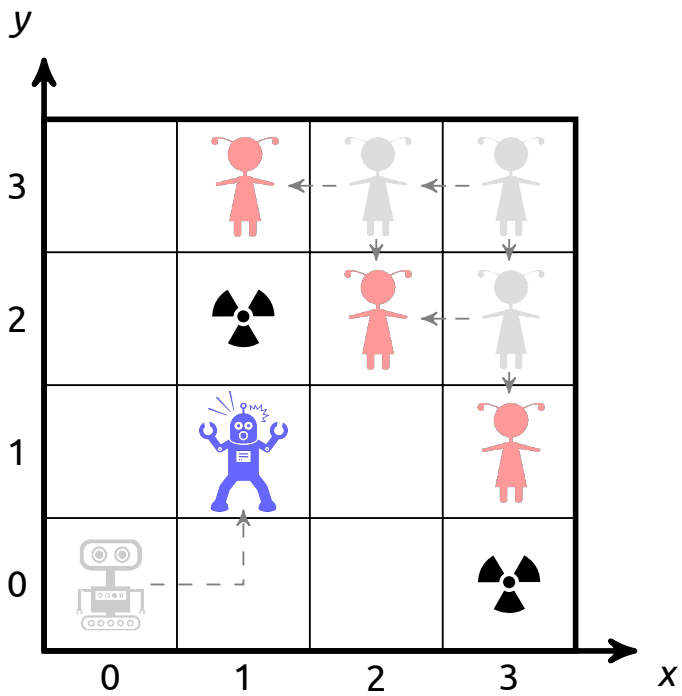


Figure – A robot escape game in a 4×4 room.

No delay :

Robot always wins by circling around the obstacle ☢ at (1,2).

1 step delay :

Robot wins by 1-step pre-decision.

2 steps delay :

Robot still wins, yet extra memory is needed.

Q2 : Do Delays Have Observable Effects?

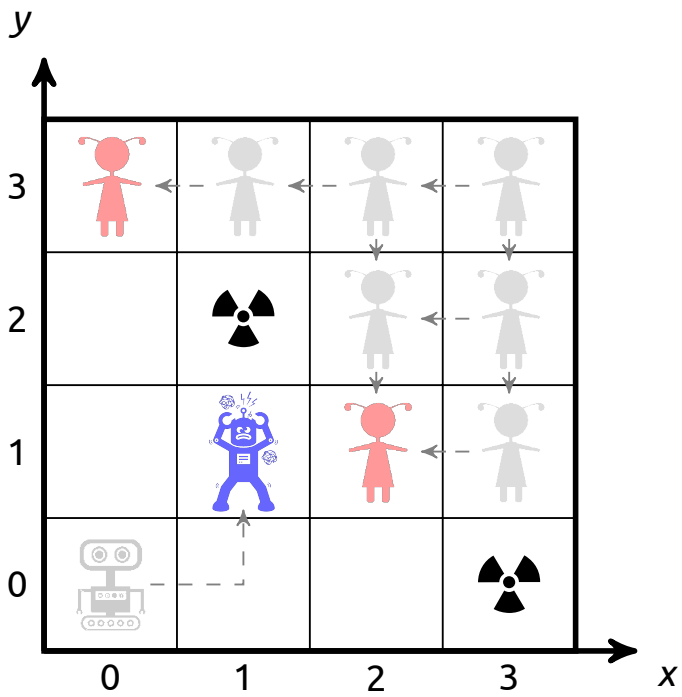


Figure – A robot escape game in a 4×4 room.

No delay :
Robot always wins by circling around the obstacle ☢ at (1,2).

1 step delay :
Robot wins by 1-step pre-decision.

2 steps delay :
Robot still wins, yet extra memory is needed.

3 steps delay :

Q2 : Do Delays Have Observable Effects?

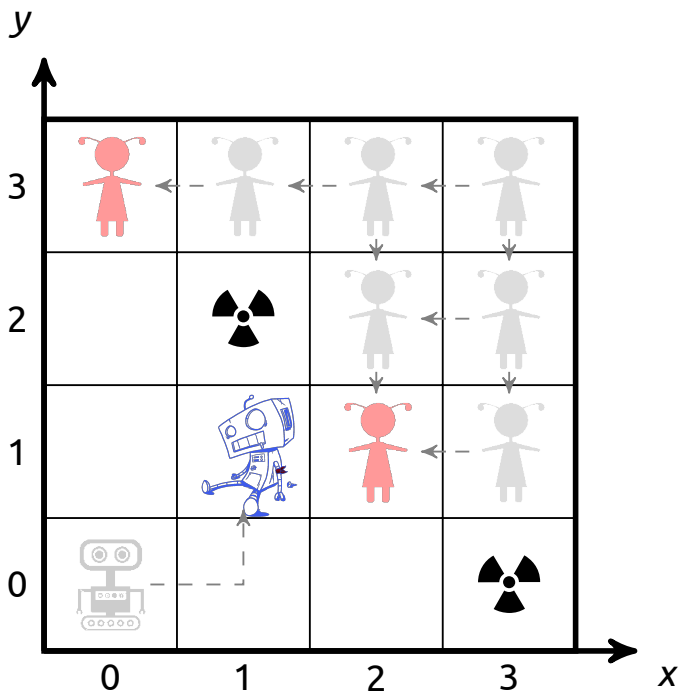


Figure – A robot escape game in a 4×4 room.

No delay :
Robot always wins by circling around the obstacle ☢ at (1,2).

1 step delay :
Robot wins by 1-step pre-decision.

2 steps delay :
Robot still wins, yet extra memory is needed.

3 steps delay :
Robot is unwinnable (uncontrollable) anymore.

Q2 : Do Delays Have Observable Effects? – Yes, they have.

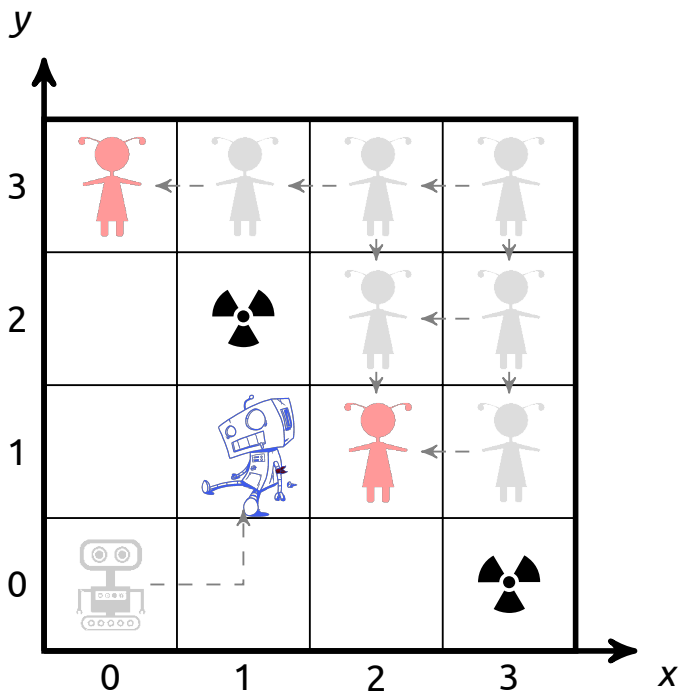


Figure – A robot escape game in a 4×4 room.

No delay :
Robot always wins by circling around the obstacle ☢ at (1,2).

1 step delay :
Robot wins by 1-step pre-decision.

2 steps delay :
Robot still wins, yet extra memory is needed.

3 steps delay :
Robot is unwinnable (uncontrollable) anymore.

Q3 : May the Effects be Harmful? – Yes, delays may well annihilate the control performance.

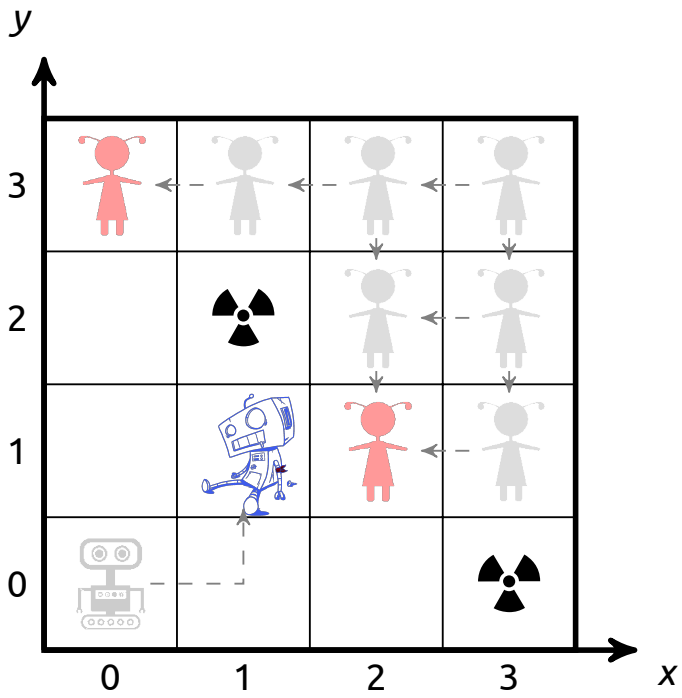


Figure – A robot escape game in a 4×4 room.

No delay :
Robot always wins by circling around the obstacle ☢ at (1,2).

1 step delay :
Robot wins by 1-step pre-decision.

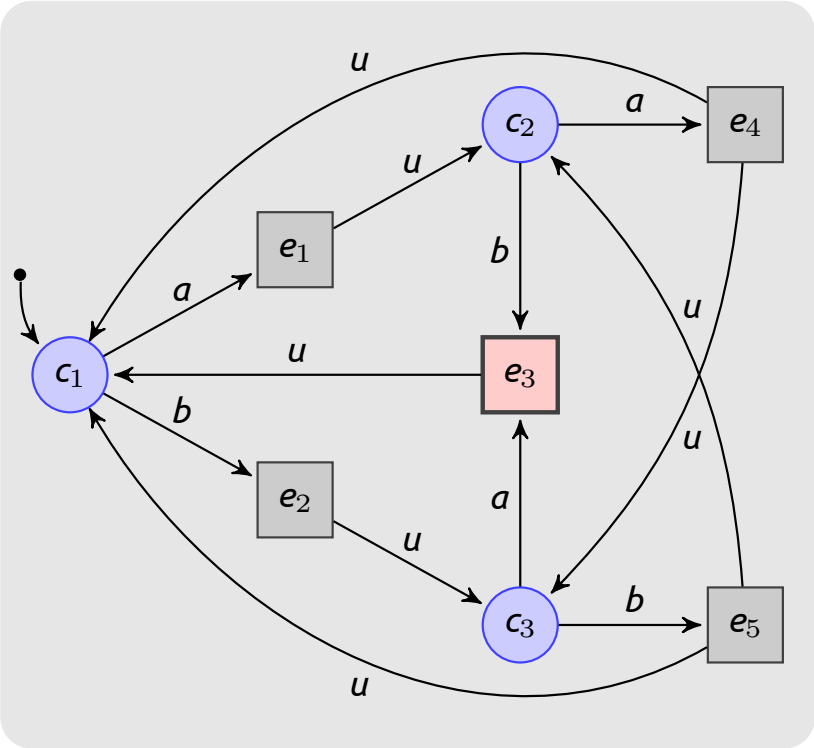
2 steps delay :
Robot still wins, yet extra memory is needed.

3 steps delay :
Robot is unwinnable (uncontrollable) anymore.

Outline

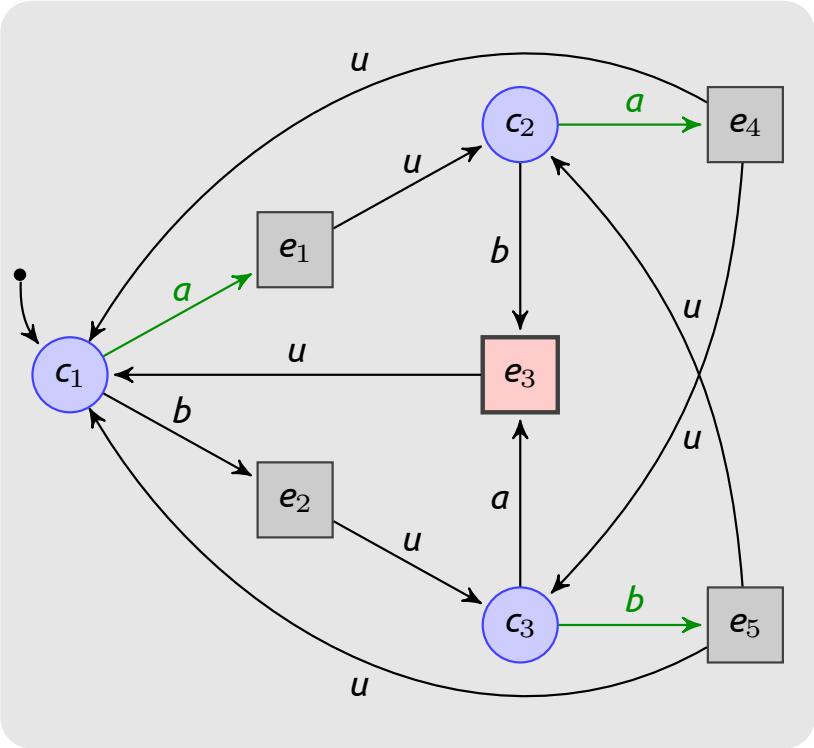
1. Safety Games under Delayed Interaction
2. Synthesizing Delay-Resilient Control
3. Extended Delay Patterns

A Trivial Safety Game



Goal : Avoid e_3 by appropriate actions of player c.

A Trivial Safety Game

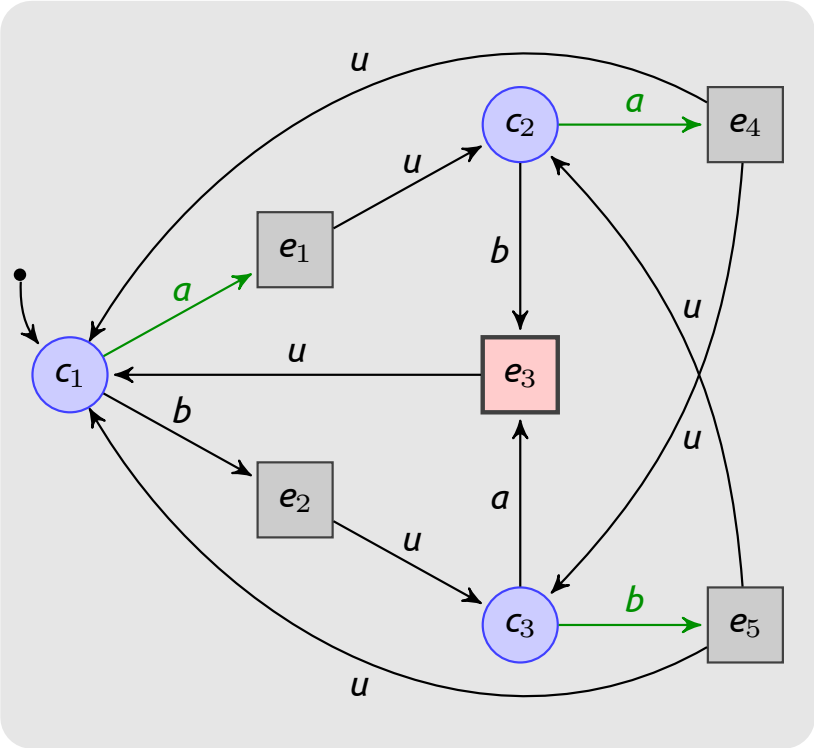


Goal : Avoid e_3 by appropriate actions of player c.

Strategy : May always play a except in c_3 :

$$c_1, c_2 \mapsto a$$
$$c_3 \mapsto b$$

A Trivial Safety Game



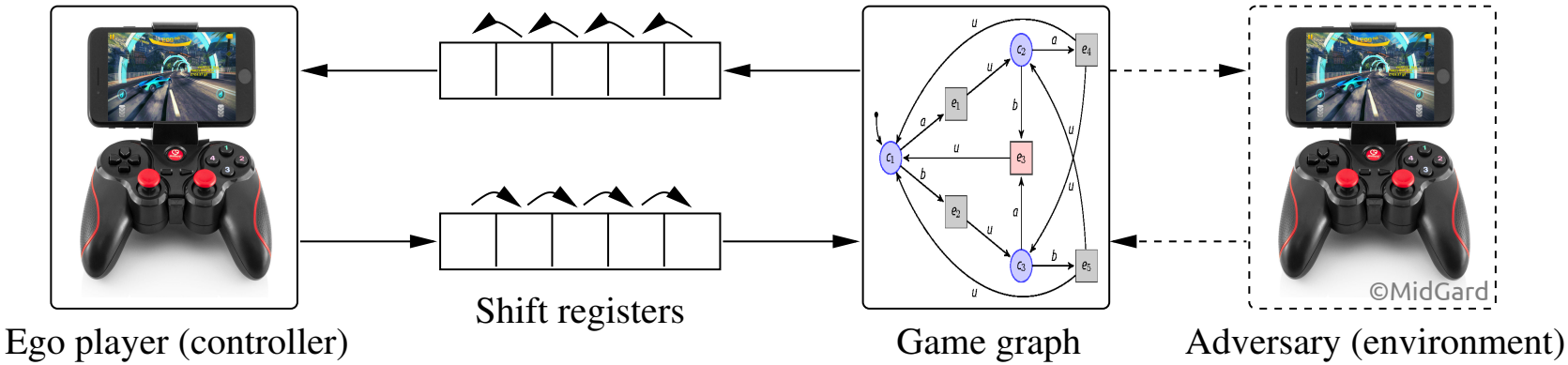
Goal : Avoid e_3 by appropriate actions of player *c*.

Strategy : May always play *a* except in c_3 :

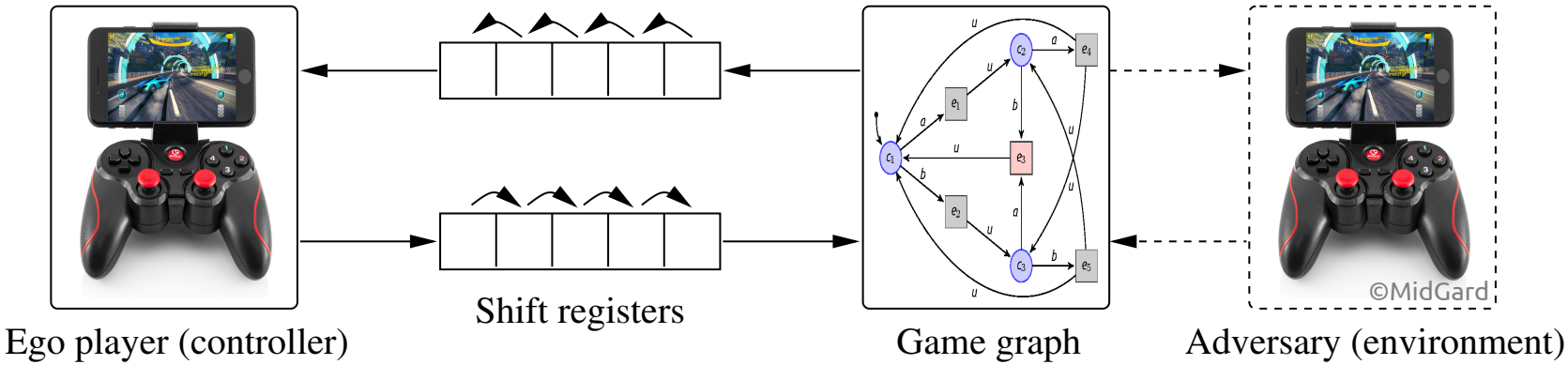
$$c_1, c_2 \mapsto a$$
$$c_3 \mapsto b$$

Properties : Determinacy and memoryless.

Playing Safety Games under Discrete Delay

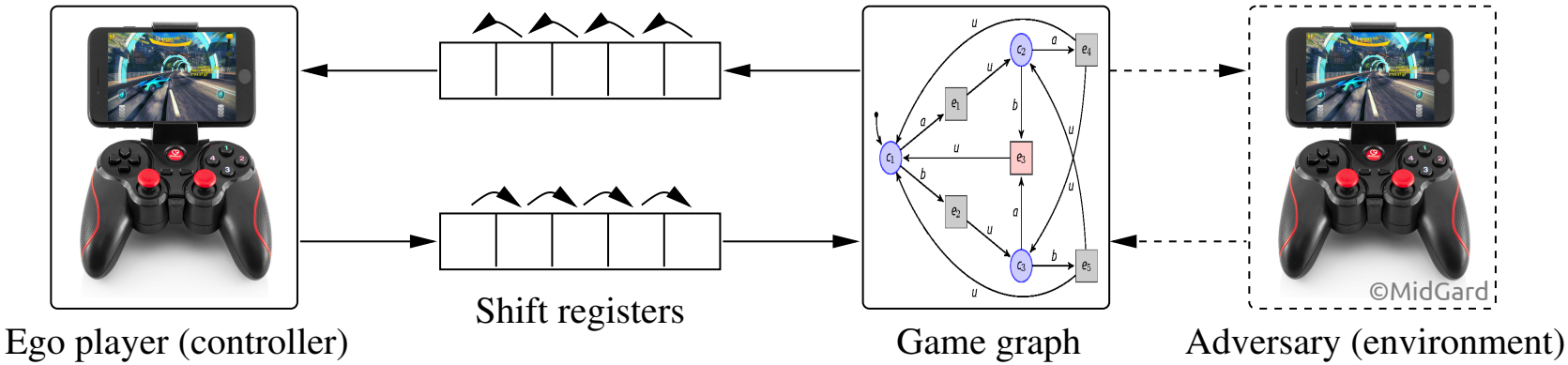


Playing Safety Games under Discrete Delay



Observation : It doesn't make an observable difference for the joint dynamics whether delay occurs in *perception*, *actuation*, or *both*.

Playing Safety Games under Discrete Delay

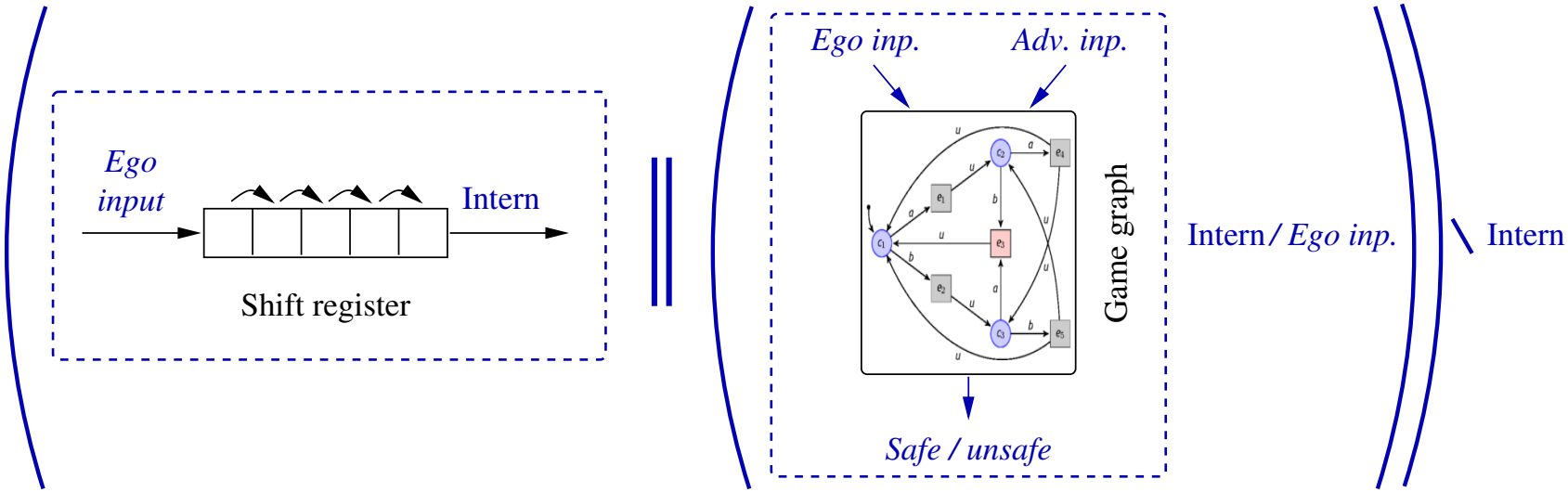


Observation : It doesn't make an observable difference for the joint dynamics whether delay occurs in *perception*, *actuation*, or *both*.

Consequence : An obvious *reduction* to a safety game of *perfect information*.

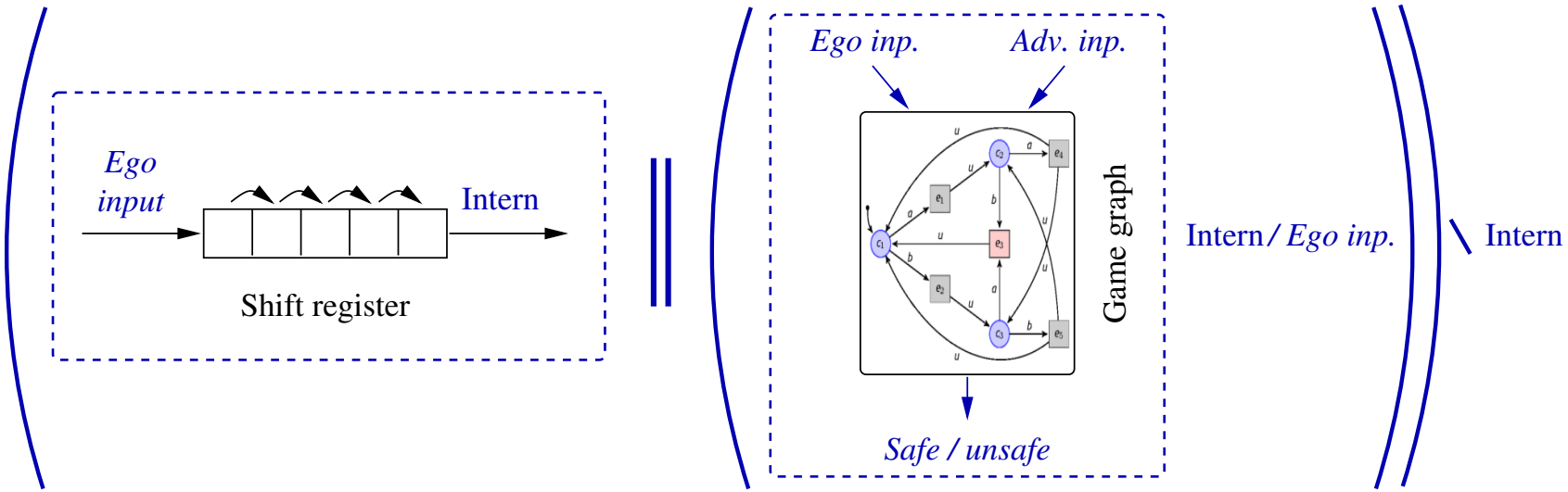
Reduction to Delay-Free Games

from Ego-Player Perspective



Reduction to Delay-Free Games

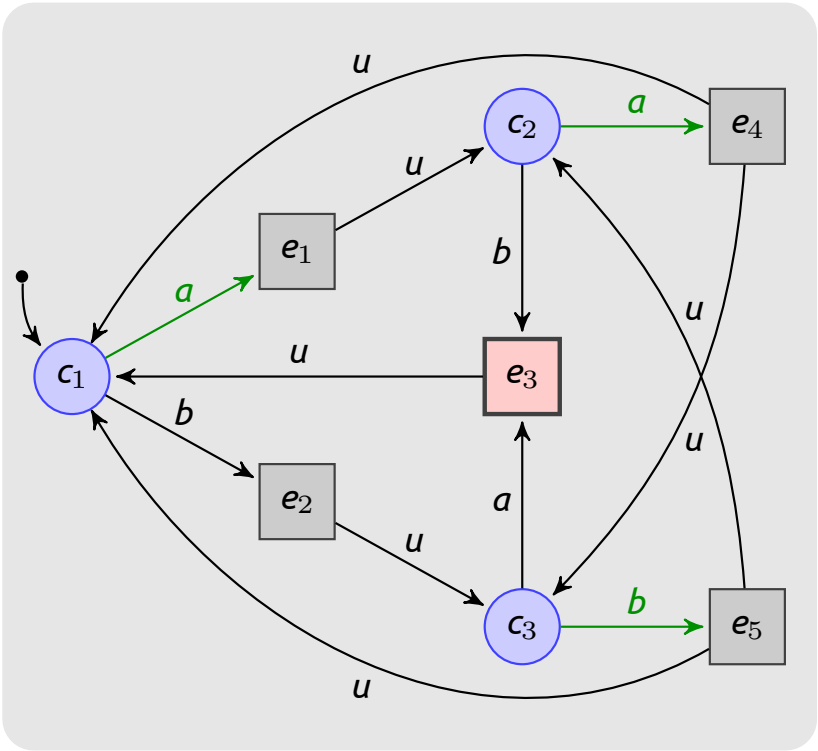
from Ego-Player Perspective



- ☺ Safety games w. delay can be solved algorithmically.
- ☹ Game graph incurs blow-up by factor $|\text{Alphabet(ego)}|^{\text{delay}}$.

The Simple Safety Game

… but with Delay

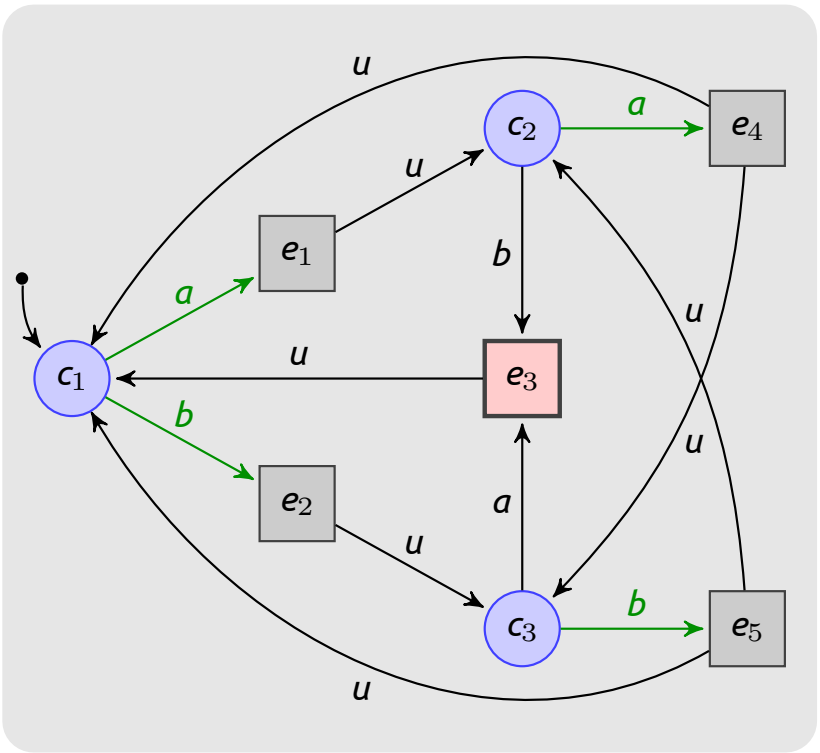


No delay :

$$c_1, c_2 \mapsto a$$
$$c_3 \mapsto b$$

The Simple Safety Game

… but with Delay



No delay :

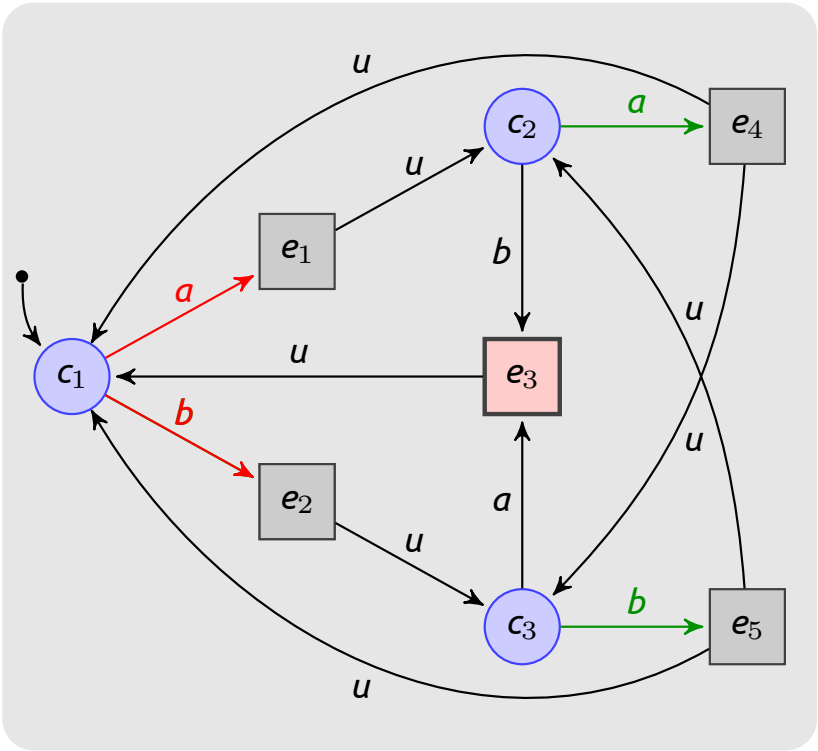
$$c_1, c_2 \mapsto a$$
$$c_3 \mapsto b$$

1 step delay :

$$e_1, e_5 \mapsto a$$
$$e_2, e_4 \mapsto b$$

The Simple Safety Game

… but with Delay



No delay :

$$c_1, c_2 \mapsto a$$
$$c_3 \mapsto b$$

1 step delay :

$$e_1, e_5 \mapsto a$$
$$e_2, e_4 \mapsto b$$

2 steps delay :

$$c_1 \mapsto \begin{cases} a & \text{if 2 steps back an } a \text{ was issued,} \\ b & \text{if 2 steps back a } b \text{ was issued.} \end{cases}$$
$$c_2 \mapsto b$$
$$c_3 \mapsto a$$

Need memory!

Incremental Synthesis in a Nutshell

Observation : A winning strategy for delay $k' > k$ can always be utilized for a safe win under delay k.

Consequence : A position is winning for delay k is a necessary condition for it being winning under delay $k' > k$.

⇒ M. Chen, M. Fränzle, Y. Li, P. N. Mosaad, N. Zhan : *What's to come is still unsure : Synthesizing controllers resilient to delayed interaction*. ATVA '18. [Distinguished Paper Award].

Incremental Synthesis in a Nutshell

Observation : A winning strategy for delay $k' > k$ can always be utilized for a safe win under delay k.

Consequence : A position is winning for delay k is a necessary condition for it being winning under delay $k' > k$.

Idea : Incrementally filter out loss states &
incrementally synthesize winning strategy for the remaining :

1. synthesize winning strategy for the *delay-free* counterpart;
2. for each winning state, *lift strategy from delay k to* $k + 1$;
3. *remove states* where this does not succeed;
4. repeat from 2 until either delay-resilience suffices (*winning*) or initial state turns lossy (*losing*).

⇒ M. Chen, M. Fränzle, Y. Li, P. N. Mosaad, N. Zhan : *What's to come is still unsure : Synthesizing controllers resilient to delayed interaction*. ATVA '18. [Distinguished Paper Award].

Incremental Synthesis in a Nutshell

Observation : A winning strategy for delay $k' > k$ can always be utilized for a safe win under delay k.

Consequence : A position is winning for delay k is a necessary condition for it being winning under delay $k' > k$.

Idea : Incrementally filter out loss states &
incrementally synthesize winning strategy for the remaining :

1. synthesize winning strategy for the *delay-free* counterpart;
2. for each winning state, *lift strategy from delay k to* $k + 1$;
3. *remove states* where this does not succeed;
4. repeat from 2 until either delay-resilience suffices (*winning*) or initial state turns lossy (*losing*).

⇒ M. Chen, M. Fränzle, Y. Li, P. N. Mosaad, N. Zhan : *What's to come is still unsure : Synthesizing controllers resilient to delayed interaction*. ATVA '18. [Distinguished Paper Award].

Incremental Synthesis in a Nutshell

Observation : A winning strategy for delay $k' > k$ can always be utilized for a safe win under delay k.

Consequence : A position is winning for delay k is a necessary condition for it being winning under delay $k' > k$.

Idea : Incrementally filter out loss states &
incrementally synthesize winning strategy for the remaining :

1. synthesize winning strategy for the *delay-free* counterpart;
2. for each winning state, *lift strategy from delay k to* $k+1$;
3. *remove states* where this does not succeed;
4. repeat from 2 until either delay-resilience suffices (*winning*) or initial state turns lossy (*losing*).

⇒ M. Chen, M. Fränzle, Y. Li, P. N. Mosaad, N. Zhan : *What's to come is still unsure : Synthesizing controllers resilient to delayed interaction*. ATVA '18. [Distinguished Paper Award].

Incremental Synthesis in a Nutshell

Observation : A winning strategy for delay $k' > k$ can always be utilized for a safe win under delay k.

Consequence : A position is winning for delay k is a necessary condition for it being winning under delay $k' > k$.

Idea : Incrementally filter out loss states &
incrementally synthesize winning strategy for the remaining :

1. synthesize winning strategy for the *delay-free* counterpart;
2. for each winning state, *lift strategy from delay k to* $k+1$;
3. *remove states* where this does not succeed;
4. repeat from 2 until either delay-resilience suffices (*winning*) or initial state turns lossy (*losing*).

⇒ M. Chen, M. Fränzle, Y. Li, P. N. Mosaad, N. Zhan : *What's to come is still unsure : Synthesizing controllers resilient to delayed interaction*. ATVA '18. [Distinguished Paper Award].

Incremental Synthesis in a Nutshell

Observation : A winning strategy for delay $k' > k$ can always be utilized for a safe win under delay k.

Consequence : A position is winning for delay k is a necessary condition for it being winning under delay $k' > k$.

Idea : Incrementally filter out loss states &
incrementally synthesize winning strategy for the remaining :

1. synthesize winning strategy for the *delay-free* counterpart;
2. for each winning state, *lift strategy from delay k to* $k+1$;
3. *remove states* where this does not succeed;
4. repeat from 2 until either delay-resilience suffices (*winning*) or initial state turns lossy (*losing*).

⇒ M. Chen, M. Fränzle, Y. Li, P. N. Mosaad, N. Zhan : *What's to come is still unsure : Synthesizing controllers resilient to delayed interaction*. ATVA '18. [Distinguished Paper Award].

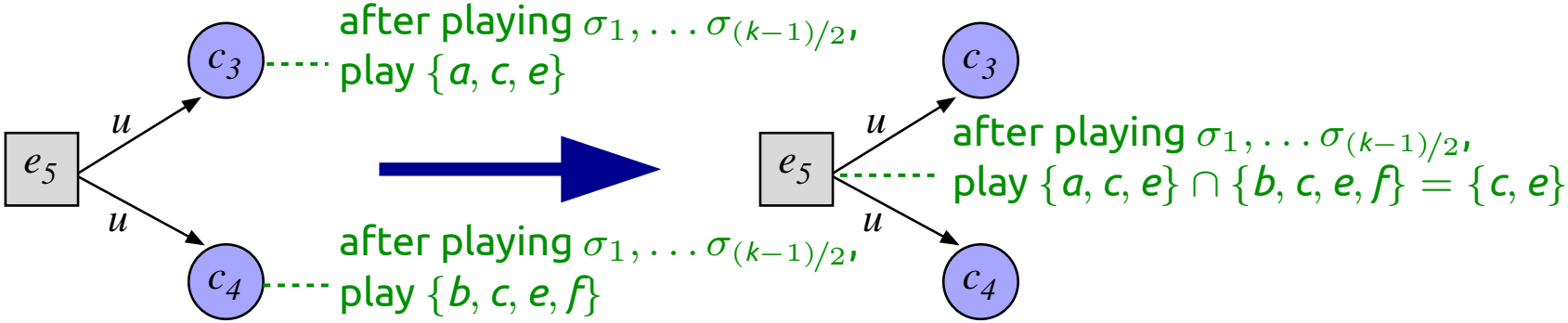
Incremental Synthesis of Delay-Tolerant Strategies

1. Generate a *maximally permissive* strategy for delay $k = 0$.

Incremental Synthesis of Delay-Tolerant Strategies

1. Generate a *maximally permissive* strategy for delay $k = 0$.
2. Advance to delay $k + 1$:

   If k odd : For each (ego-)winning adversarial state define strategy as

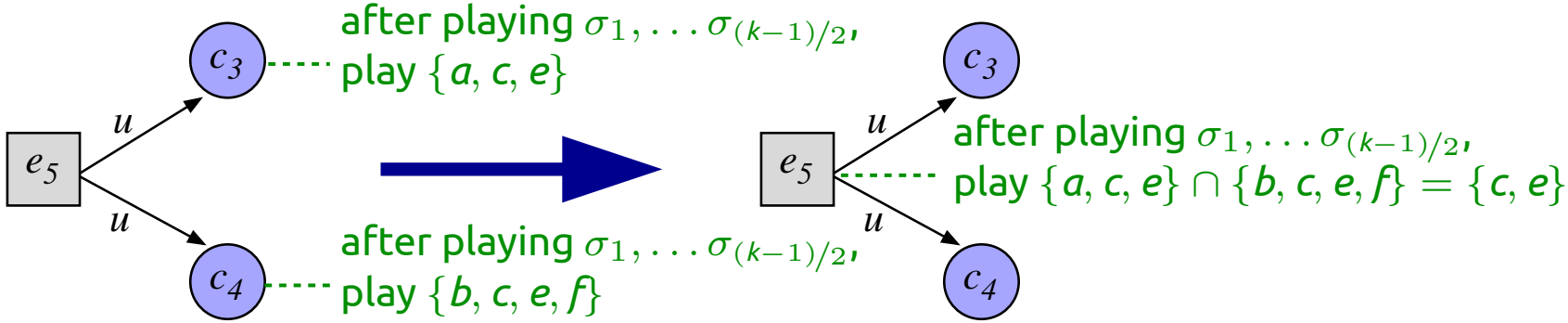


… and eliminate any dead ends by bwd. traversal.

Incremental Synthesis of Delay-Tolerant Strategies

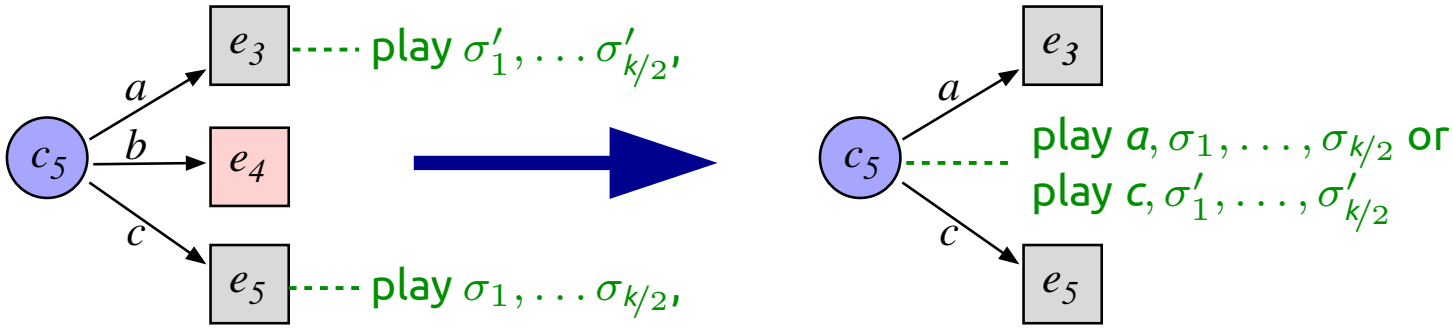
1. Generate a *maximally permissive* strategy for delay $k = 0$.
2. Advance to delay $k + 1$:

   If k odd : For each (ego-)winning adversarial state define strategy as



… and eliminate any dead ends by bwd. traversal.

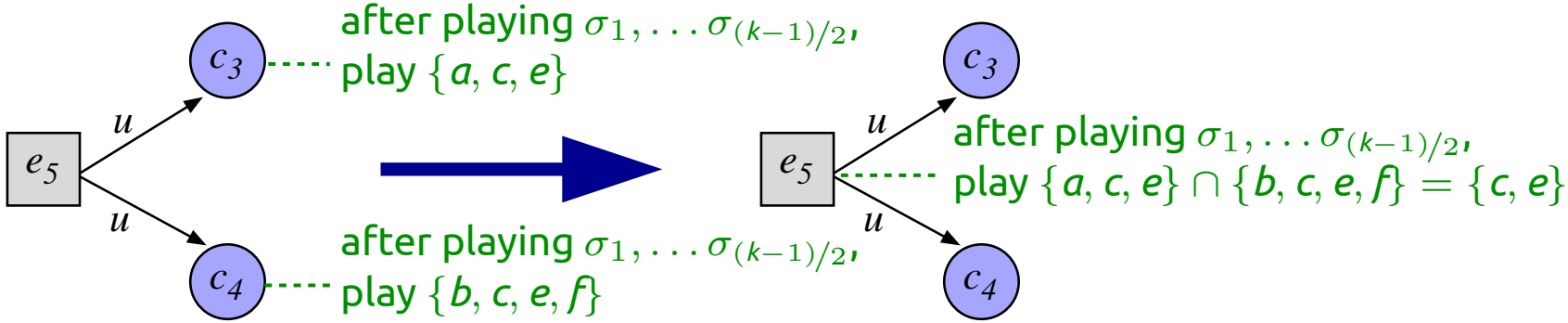
If k even : For each winning ego state define strategy as



Incremental Synthesis of Delay-Tolerant Strategies

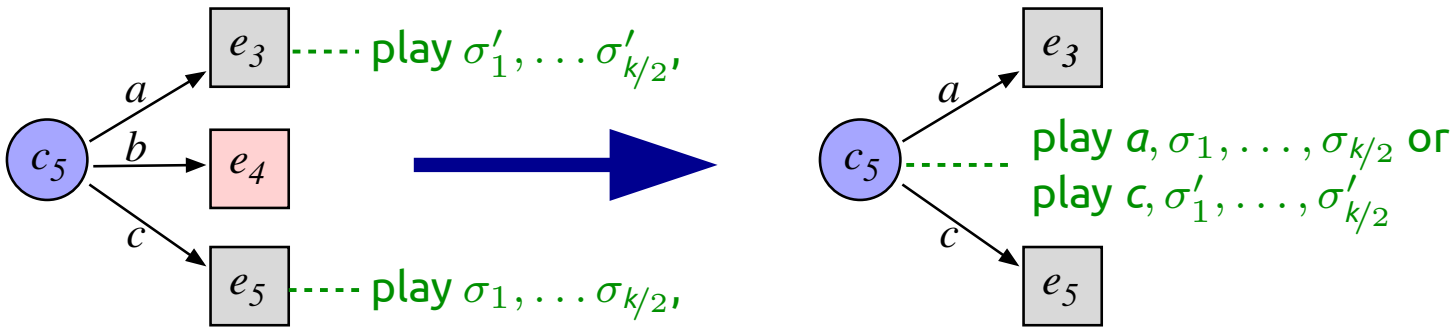
1. Generate a *maximally permissive* strategy for delay $k = 0$.
2. Advance to delay $k + 1$:

   If k odd : For each (ego-)winning adversarial state define strategy as



   … and eliminate any dead ends by bwd. traversal.

   If k even : For each winning ego state define strategy as



3. Repeat from 2 until either delay-resilience suffices or initial state turns lossy.

Incremental- vs. Reduction-Based Synthesis

Benchmark				Reduction + Explicit-State Synthesis						Incremental Explicit-State Synthesis						
name	$\lvert S\rvert$	$\lvert\rightarrow\rvert$	$\lvert\mathcal{U}\rvert$	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	%
Exmp.trv1	14	20	4	≥ 22	0.00	0.00	0.01	0.02	0.02	$\geq$ **30**	0.00	0.00	**0.00**	**0.01**	**0.01**	
Exmp.trv2	14	22	4	$=2$	0.00	0.01	0.01	0.02	–	$=2$	0.00	**0.00**	**0.00**	**0.01**	–	81.97
Escp.4×4	224	738	16	$=2$	0.08	11.66	11.73	1059.23	–	$=2$	0.08	**0.13**	**0.22**	**0.25**	–	99.02
Escp.4×5	360	1326	20	$=2$	0.18	34.09	33.80	3084.58	–	$=2$	0.18	**0.27**	**0.46**	**0.63**	–	99.02
Escp.5×5	598	2301	26	≥ 2	0.46	96.24	97.10	?	?	**=2**	0.46	**0.68**	**1.16**	**1.71**	–	98.98
Escp.5×6	840	3516	30	≥ 2	1.01	217.63	216.83	?	?	**=2**	**1.00**	**1.42**	**2.40**	**4.30**	–	99.00
Escp.6×6	1224	5424	36	≥ 2	2.13	516.92	511.41	?	?	**=2**	**2.06**	**2.90**	**5.12**	**10.30**	–	98.97
Escp.7×7	2350	11097	50	≥ 2	7.81	2167.86	2183.01	?	?	**=2**	**7.71**	**10.67**	**19.04**	**52.47**	–	98.99
Escp.7×8	3024	14820	56	≥ 0	**13.07**	?	?	?	?	**=2**	13.44	**18.25**	**32.69**	**108.60**	–	99.01

Benchmark		Reduction + Yosys + SafetySynth (symbolic)							Incremental Synthesis (explicit-state implementation)							
name	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta=5$	$\delta=6$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta=5$	$\delta=6$	%
Stub.4×4	$=2$	1.07	1.24	1.24	1.80	–	–	–	**0.04**	**0.07**	**0.12**	**0.18**	–	–	–	98.98
Stub.4×5	$=2$	1.16	1.49	1.49	2.83	–	–	–	**0.08**	**0.14**	**0.25**	**0.44**	–	–	–	98.97
Stub.5×5	$=2$	1.19	2.61	2.50	13.67	–	–	–	**0.21**	**0.37**	**0.63**	**1.17**	–	–	–	98.97
Stub.5×6	$=2$	1.18	2.60	2.59	23.30	–	–	–	**0.42**	**0.69**	**1.20**	**2.49**	–	–	–	98.96
Stub.6×6	$=4$	1.17	2.76	2.74	19.96	19.69	655.24	–	**0.93**	**1.47**	**2.60**	**5.79**	**7.54**	**7.60**	–	99.89
Stub.7×7	$=4$	**1.23**	**2.50**	**2.48**	24.57	**23.01**	2224.62	–	3.60	5.52	10.08	**22.75**	31.18	**32.98**	–	99.88

Incremental- vs. Reduction-Based Synthesis

Benchmark				Reduction + Explicit-State Synthesis						Incremental Explicit-State Synthesis						
name	$\|S\|$	$\|\rightarrow\|$	$\|\mathcal{U}\|$	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	%
Exmp.trv1	14	20	4	≥ 22	0.00	0.00	0.01	0.02	0.02	$\geq$ **30**	0.00	0.00	**0.00**	**0.01**	**0.01**	
Exmp.trv2	14	22	4	$=2$	0.00	0.01	0.01	0.02	–	$=2$	0.00	**0.00**	**0.00**	**0.01**	–	81.97
Escp.4×4	224	738	16	$=2$	0.08	11.66	11.73	1059.23	–	$=2$	0.08	**0.13**	**0.22**	**0.25**	–	99.02
Escp.4×5	360	1326	20	$=2$	0.18	34.09	33.80	3084.58	–	$=2$	0.18	**0.27**	**0.46**	**0.63**	–	99.02
Escp.5×5	598	2301	26	≥ 2	0.46	96.24	97.10	?	?	**=2**	0.46	**0.68**	**1.16**	**1.71**	–	98.98
Escp.5×6	840	3516	30	≥ 2	1.01	217.63	216.83	?	?	**=2**	**1.00**	**1.42**	**2.40**	**4.30**	–	99.00
Escp.6×6	1224	5424	36	≥ 2	2.13	516.92	511.41	?	?	**=2**	**2.06**	**2.90**	**5.12**	**10.30**	–	98.97
Escp.7×7	2350	11097	50	≥ 2	7.81	2167.86	2183.01	?	?	**=2**	**7.71**	**10.67**	**19.04**	**52.47**	–	98.99
Escp.7×8	3024	14820	56	≥ 0	**13.07**	?	?	?	?	**=2**	13.44	**18.25**	**32.69**	**108.60**	–	99.01

Benchmark		Reduction + Yosys + SafetySynth (symbolic)							Incremental Synthesis (explicit-state implementation)							
name	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta=5$	$\delta=6$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta=5$	$\delta=6$	%
Stub.4×4	$=2$	1.07	1.24	1.24	1.80	–	–	–	**0.04**	**0.07**	**0.12**	**0.18**	–	–	–	98.98
Stub.4×5	$=2$	1.16	1.49	1.49	2.83	–	–	–	**0.08**	**0.14**	**0.25**	**0.44**	–	–	–	98.97
Stub.5×5	$=2$	1.19	2.61	2.50	13.67	–	–	–	**0.21**	**0.37**	**0.63**	**1.17**	–	–	–	98.97
Stub.5×6	$=2$	1.18	2.60	2.59	23.30	–	–	–	**0.42**	**0.69**	**1.20**	**2.49**	–	–	–	98.96
Stub.6×6	$=4$	1.17	2.76	2.74	19.96	19.69	655.24	–	**0.93**	**1.47**	**2.60**	**5.79**	**7.54**	**7.60**	–	99.89
Stub.7×7	$=4$	**1.23**	**2.50**	**2.48**	24.57	**23.01**	2224.62	–	3.60	5.52	10.08	**22.75**	31.18	**32.98**	–	99.88

Incremental- vs. Reduction-Based Synthesis

Benchmark				Reduction + Explicit-State Synthesis						Incremental Explicit-State Synthesis						
name	$\|S\|$	$\|\rightarrow\|$	$\|\mathcal{U}\|$	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	%
Exmp.trv1	14	20	4	≥ 22	0.00	0.00	0.01	0.02	0.02	$\geq$ **30**	0.00	0.00	**0.00**	**0.01**	**0.01**	
Exmp.trv2	14	22	4	$=2$	0.00	0.01	0.01	0.02	–	$=2$	0.00	**0.00**	**0.00**	**0.01**	–	81.97
Escp.4×4	224	738	16	$=2$	0.08	11.66	11.73	1059.23	–	$=2$	0.08	**0.13**	**0.22**	**0.25**	–	99.02
Escp.4×5	360	1326	20	$=2$	0.18	34.09	33.80	3084.58	–	$=2$	0.18	**0.27**	**0.46**	**0.63**	–	99.02
Escp.5×5	598	2301	26	≥ 2	0.46	96.24	97.10	?	?	**=2**	0.46	**0.68**	**1.16**	**1.71**	–	98.98
Escp.5×6	840	3516	30	≥ 2	1.01	217.63	216.83	?	?	**=2**	**1.00**	**1.42**	**2.40**	**4.30**	–	99.00
Escp.6×6	1224	5424	36	≥ 2	2.13	516.92	511.41	?	?	**=2**	**2.06**	**2.90**	**5.12**	**10.30**	–	98.97
Escp.7×7	2350	11097	50	≥ 2	7.81	2167.86	2183.01	?	?	**=2**	**7.71**	**10.67**	**19.04**	**52.47**	–	98.99
Escp.7×8	3024	14820	56	≥ 0	**13.07**	?	?	?	?	**=2**	13.44	**18.25**	**32.69**	**108.60**	–	99.01

Benchmark		Reduction + Yosys + SafetySynth (symbolic)							Incremental Synthesis (explicit-state implementation)							
name	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta=5$	$\delta=6$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta=5$	$\delta=6$	%
Stub.4×4	$=2$	1.07	1.24	1.24	1.80	–	–	–	**0.04**	**0.07**	**0.12**	**0.18**	–	–	–	98.98
Stub.4×5	$=2$	1.16	1.49	1.49	2.83	–	–	–	**0.08**	**0.14**	**0.25**	**0.44**	–	–	–	98.97
Stub.5×5	$=2$	1.19	2.61	2.50	13.67	–	–	–	**0.21**	**0.37**	**0.63**	**1.17**	–	–	–	98.97
Stub.5×6	$=2$	1.18	2.60	2.59	23.30	–	–	–	**0.42**	**0.69**	**1.20**	**2.49**	–	–	–	98.96
Stub.6×6	$=4$	1.17	2.76	2.74	19.96	19.69	655.24	–	**0.93**	**1.47**	**2.60**	**5.79**	**7.54**	**7.60**	–	99.89
Stub.7×7	$=4$	**1.23**	**2.50**	**2.48**	24.57	**23.01**	2224.62	–	3.60	5.52	10.08	**22.75**	31.18	**32.98**	–	99.88

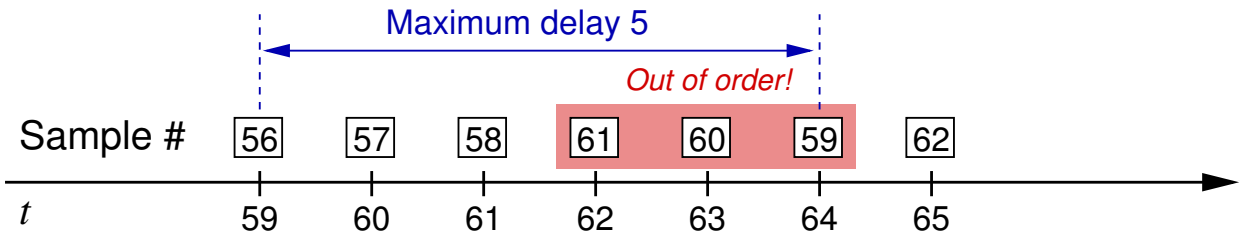
Incremental- vs. Reduction-Based Synthesis

Benchmark				Reduction + Explicit-State Synthesis						Incremental Explicit-State Synthesis						
name	$\|S\|$	$\|\rightarrow\|$	$\|\mathcal{U}\|$	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	%
Exmp.trv1	14	20	4	≥ 22	0.00	0.00	0.01	0.02	0.02	$\geq$ **30**	0.00	0.00	**0.00**	**0.01**	**0.01**	
Exmp.trv2	14	22	4	$=2$	0.00	0.01	0.01	0.02	–	$=2$	0.00	**0.00**	**0.00**	**0.01**	–	81.97
Escp.4×4	224	738	16	$=2$	0.08	11.66	11.73	1059.23	–	$=2$	0.08	**0.13**	**0.22**	**0.25**	–	99.02
Escp.4×5	360	1326	20	$=2$	0.18	34.09	33.80	3084.58	–	$=2$	0.18	**0.27**	**0.46**	**0.63**	–	99.02
Escp.5×5	598	2301	26	≥ 2	0.46	96.24	97.10	?	?	**=2**	0.46	**0.68**	**1.16**	**1.71**	–	98.98
Escp.5×6	840	3516	30	≥ 2	1.01	217.63	216.83	?	?	**=2**	**1.00**	**1.42**	**2.40**	**4.30**	–	99.00
Escp.6×6	1224	5424	36	≥ 2	2.13	516.92	511.41	?	?	**=2**	**2.06**	**2.90**	**5.12**	**10.30**	–	98.97
Escp.7×7	2350	11097	50	≥ 2	7.81	2167.86	2183.01	?	?	**=2**	**7.71**	**10.67**	**19.04**	**52.47**	–	98.99
Escp.7×8	3024	14820	56	≥ 0	**13.07**	?	?	?	?	**=2**	13.44	**18.25**	**32.69**	**108.60**	–	99.01

Benchmark		Reduction + Yosys + SafetySynth (symbolic)							Incremental Synthesis (explicit-state implementation)							
name	$\delta_{\max}$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta=5$	$\delta=6$	$\delta=0$	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=4$	$\delta=5$	$\delta=6$	%
Stub.4×4	$=2$	1.07	1.24	1.24	1.80	–	–	–	**0.04**	**0.07**	**0.12**	**0.18**	–	–	–	98.98
Stub.4×5	$=2$	1.16	1.49	1.49	2.83	–	–	–	**0.08**	**0.14**	**0.25**	**0.44**	–	–	–	98.97
Stub.5×5	$=2$	1.19	2.61	2.50	13.67	–	–	–	**0.21**	**0.37**	**0.63**	**1.17**	–	–	–	98.97
Stub.5×6	$=2$	1.18	2.60	2.59	23.30	–	–	–	**0.42**	**0.69**	**1.20**	**2.49**	–	–	–	98.96
Stub.6×6	$=4$	1.17	2.76	2.74	19.96	19.69	655.24	–	**0.93**	**1.47**	**2.60**	**5.79**	**7.54**	**7.60**	–	99.89
Stub.7×7	$=4$	**1.23**	**2.50**	**2.48**	24.57	**23.01**	2224.62	–	3.60	5.52	10.08	**22.75**	31.18	**32.98**	–	99.88

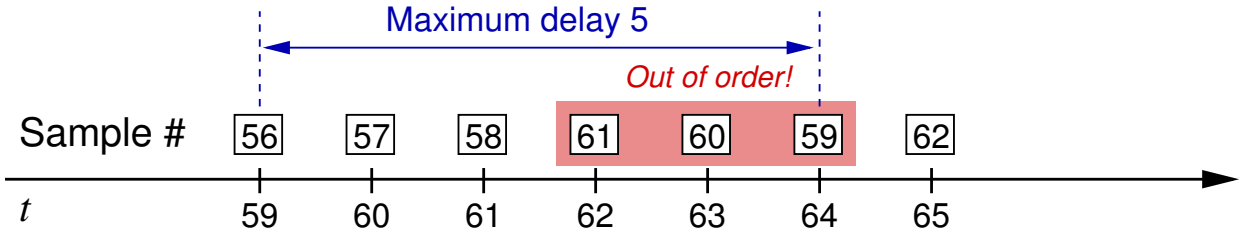
Out-of-Order Message Delivery

☹ Observations may arrive *out-of-order* :

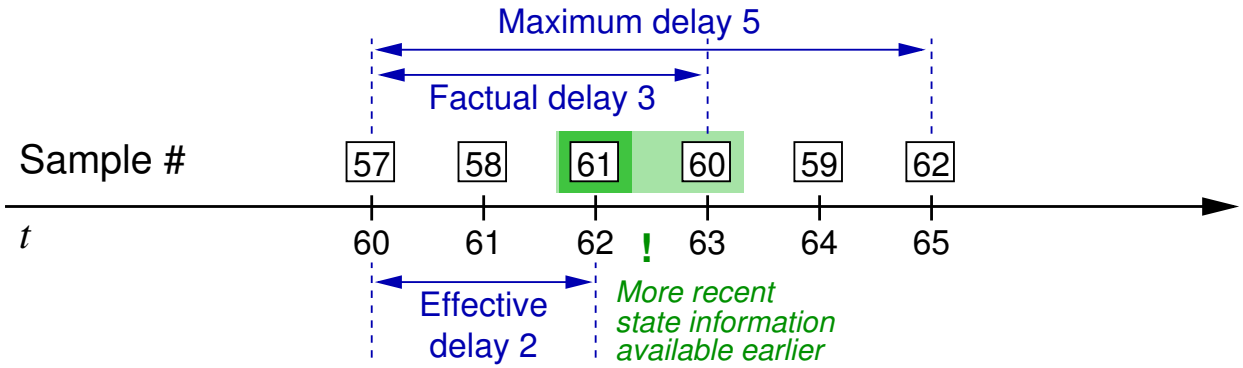


Out-of-Order Message Delivery

☹ Observations may arrive *out-of-order* :

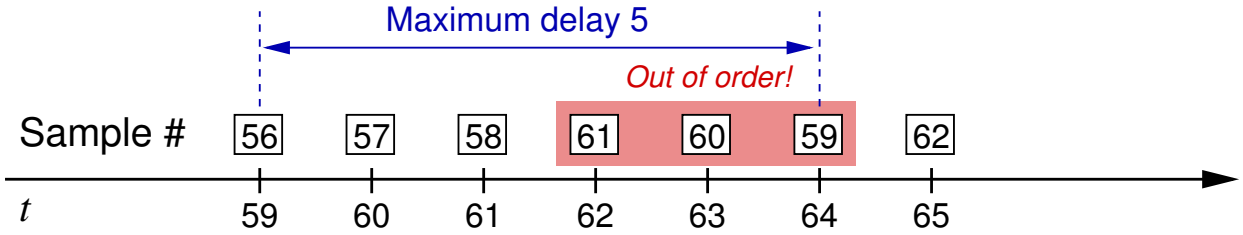


☺ But this may only reduce effective delay, improving controllability :

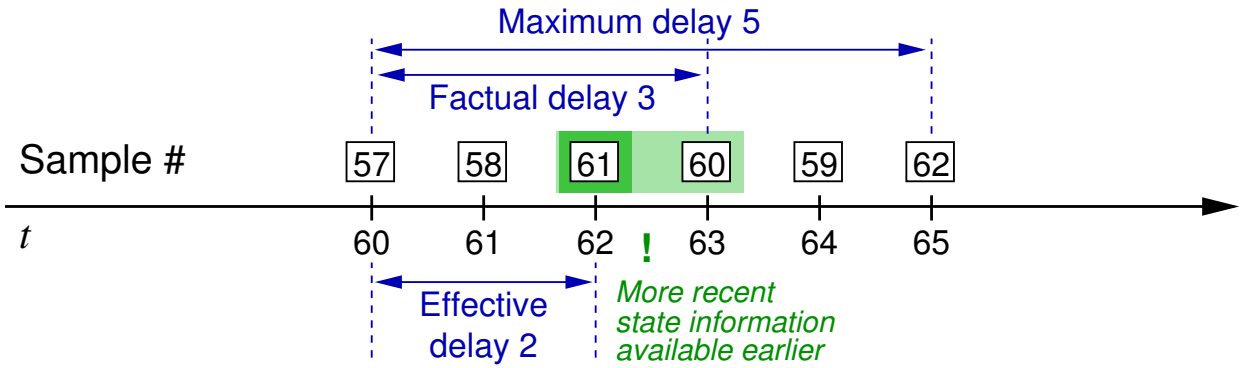


Out-of-Order Message Delivery

☹ Observations may arrive *out-of-order* :



☺ But this may only reduce effective delay, improving controllability :

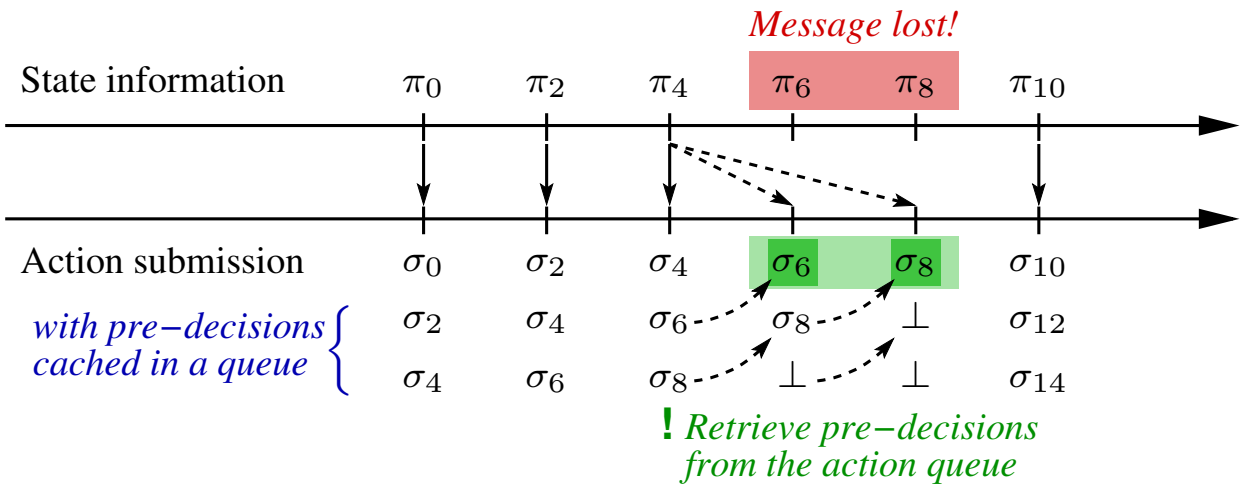


☺ W.r.t. qualitative controllability, the worst-case of out-of-order delivery is equivalent to order-preserving delay *k*.

☺ Stochastically expected controllability even better than for strict delay *k*.

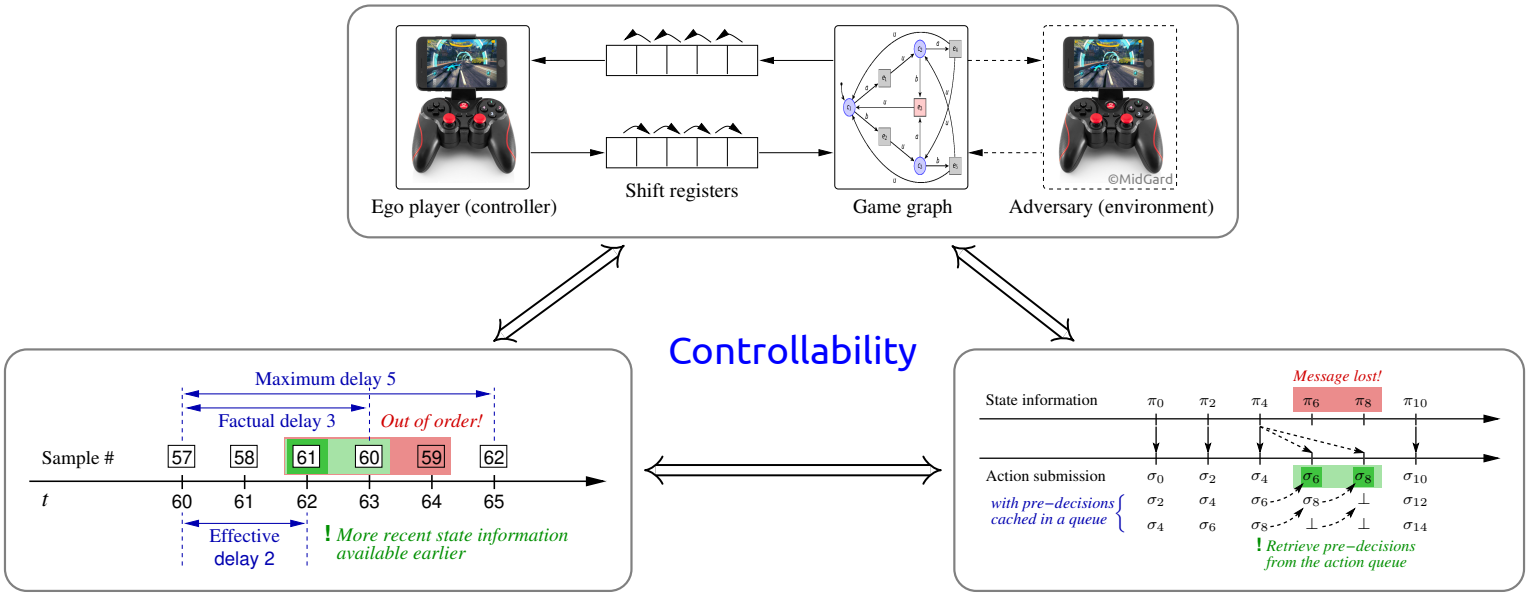
(Bounded) Message Loss

☹ Message carrying the state information may get *lost* :



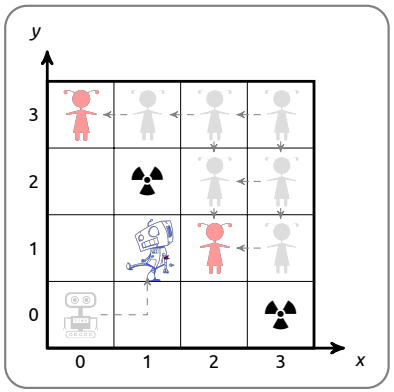
☺ The controller can still win a safety game in the presence of bounded message loss leveraging delay-resilient strategies.

Equivalence of Qualitative Controllability

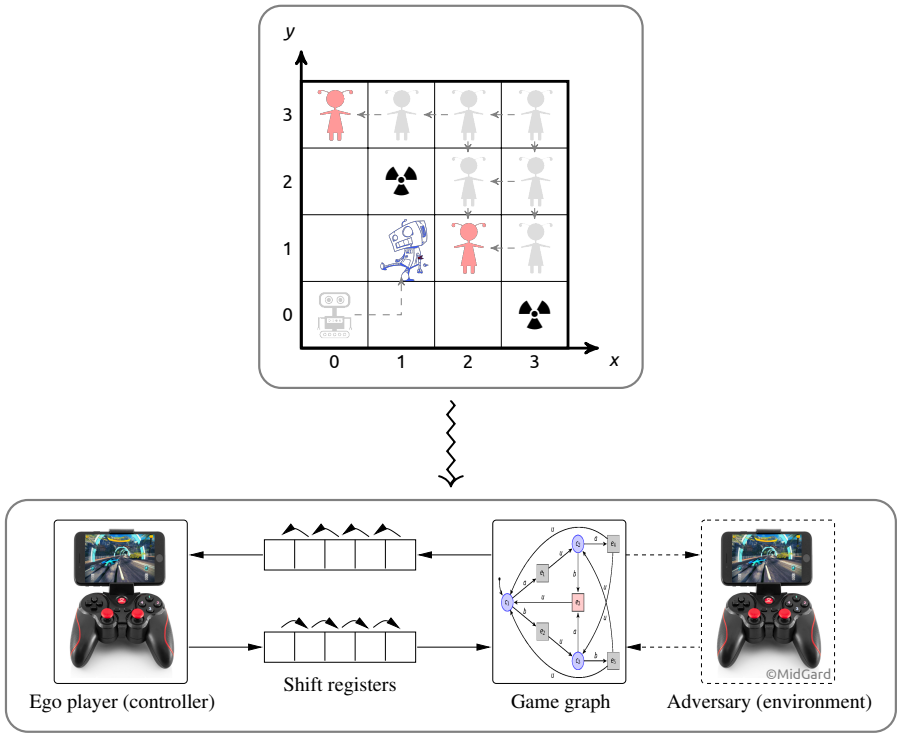


⇒ M. Chen, M. Fränzle, Y. Li, P. N. Mosaad, N. Zhan : *Indecision and delays are the parents of failure : Taming them algorithmically by synthesizing delay-resilient control.* Acta Informatica '20.

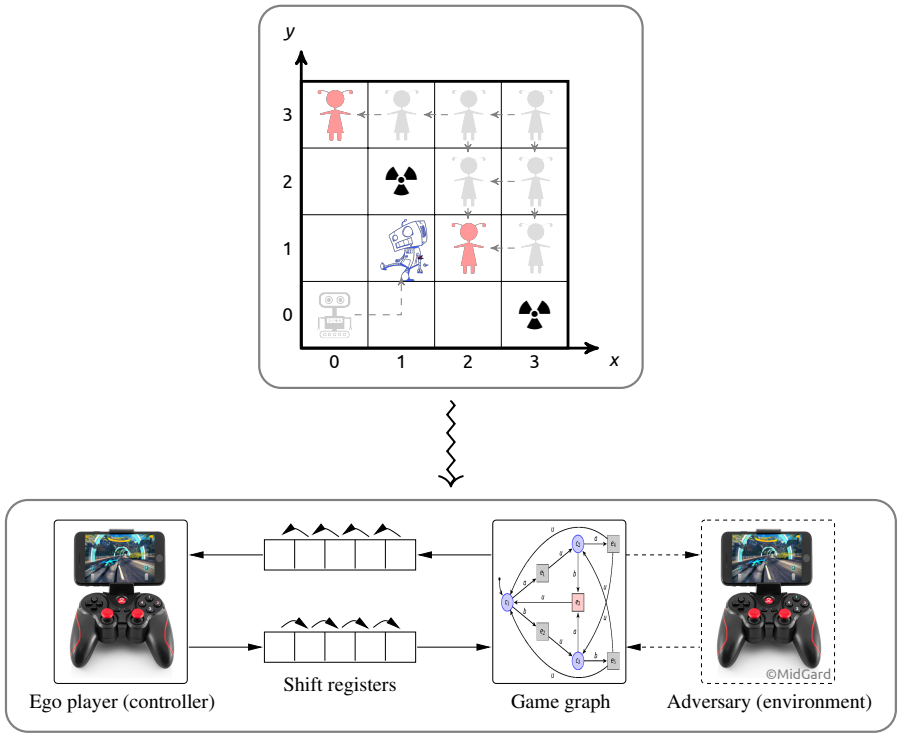
Summary



Summary

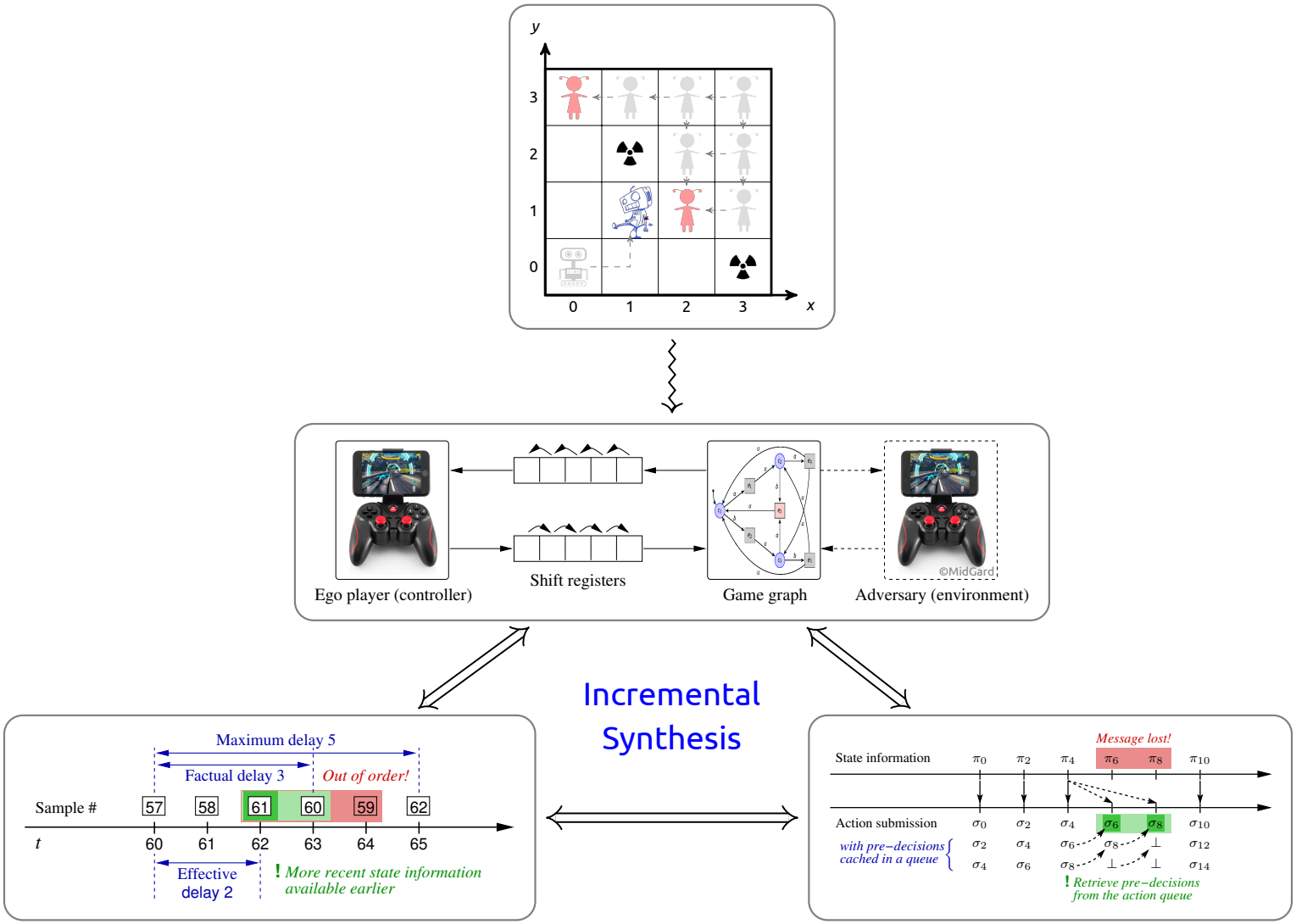

y
x
0
1
2
3
Ego player (controller)
Shift registers
Game graph
Adversary (environment)
©MidGard

Summary

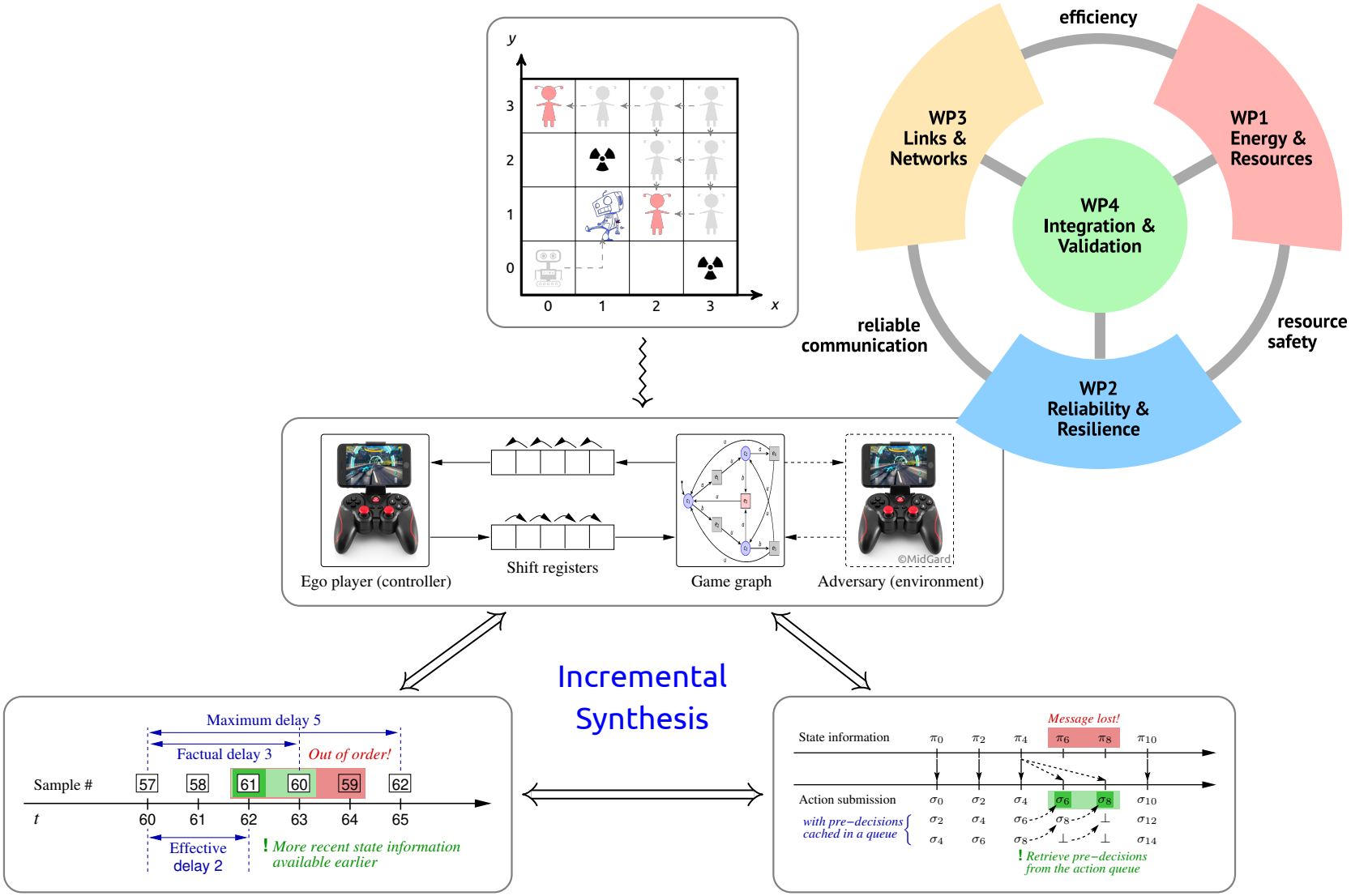


Incremental Synthesis

Summary


y
x
0
1
2
3
Ego player (controller)
Shift registers
Game graph
Adversary (environment)
©MidGard
Incremental Synthesis
Maximum delay 5
Factual delay 3
Out of order!
Sample #
57
58
61
60
59
62
t
60
61
62
63
64
65
Effective delay 2
! More recent state information available earlier
Message lost!
State information
Action submission
with pre-decisions cached in a queue
! Retrieve pre-decisions from the action queue

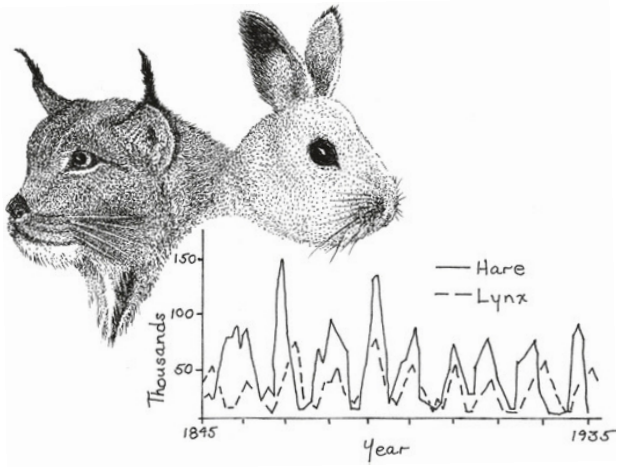
Summary


y
x
0
1
2
3
efficiency
WP3
Links &
Networks
WP1
Energy &
Resources
WP4
Integration &
Validation
reliable
communication
resource
safety
WP2
Reliability &
Resilience
Ego player (controller)
Shift registers
Game graph
Adversary (environment)
©MidGard
Incremental
Synthesis
Maximum delay 5
Factual delay 3
Out of order!
Sample #
57
58
61
60
59
62
t
60
61
62
63
64
65
Effective
delay 2
! More recent state information
available earlier
Message lost!
State information
Action submission
with pre–decisions
cached in a queue
! Retrieve pre–decisions
from the action queue

Delays in Dynamical/Hybrid Systems

©Wikipedia

Vito Volterra



©J. Pastor, 2016

Predator-prey dynamics

"[...] the rate of change of physical systems depends not only on their present state, but also on their past history."

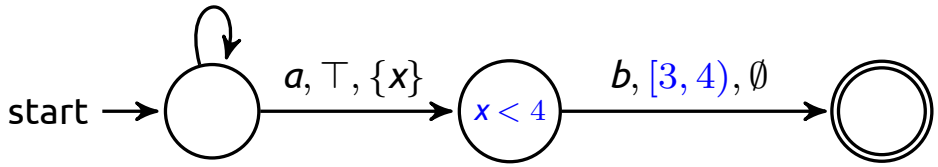
[Richard Bellman and Kenneth L. Cooke, 1963]

⇒ M. Chen, M. Fränzle, Y. Li, P. N. Mosaad, N. Zhan : *Validat. simul.-based verific. of DDEs*. FM'16.
⇒ B. Xue, P. N. Mosaad, M. Fränzle, M. Chen, Y. Li, N. Zhan : *Safe approx. of reach. sets for DDEs*. FORMATS'17.
⇒ S. Feng, M. Chen, N. Zhan, M. Fränzle, B. Xue : *Taming delays in dyn. syst. : Unbounded verif. of DDEs*. CAV'19.

Q1a : Resultant Forms of Delay

Delayed reaction : Reaction to a stimulus is not immediate.

- Easy to model in timed/hybrid automata, etc. :



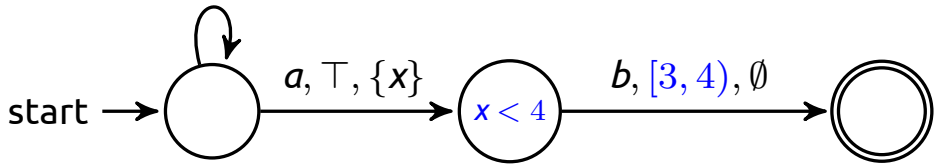
- Thus amenable to the pertinent analysis tools.

⇒ Not of interest today.

Q1a : Resultant Forms of Delay

Delayed reaction : Reaction to a stimulus is not immediate.

- Easy to model in timed/hybrid automata, etc. :



- Thus amenable to the pertinent analysis tools.

⇒ Not of interest today.

Network delay : Information of different age coexists and is queuing in the network when piped towards target.

- End-to-end latency may exceed sampling intervals etc. by orders of magnitude.
- Not (efficiently) expressible in standard models.

⇒ Our theme today : discrete-time pipelined delay.

[Chen *et al.* : ATVA '18, Acta Inf. '20];
[Zimmermann : LICS '18, GandALF '17], [Klein & Zimmermann : ICALP '15, CSL '15].

Equivalence of Qualitative Controllability

Theorem (Equivalence of qualitative controllability)

Given a two-player safety game, the following statements are equivalent if δ is even :

1. *There exists a winning strategy under an exact delay of δ, i.e., if at any point of time t the control strategy is computed based on a prefix of the game that has length $t - \delta$.*
2. *There exists a winning strategy under time-stamped out-of-order delivery with a maximum delay of δ, i.e., if at any point of time t the control strategy is computed based on the complete prefix of the game of length $t - \delta$ plus potentially available partial knowledge of the game states between $t - \delta$ and t.*
3. *There exists a winning strategy when at any time $t = 2n$, i.e., any player-0 move, information on the game state at some time $t' \in \{t - 2k, \ldots, t\}$ is available, i.e., under out-of-order delivery of messages with a maximum delay of δ and a maximum number of consecutively lost upstream or downstream messages of $\delta/2$.*

The first two equivalences do also hold for odd δ.

⇒ M. Chen, M. Fränzle, Y. Li, P. N. Mosaad, N. Zhan : *Indecision and delays are the parents of failure : Taming them algorithmically by synthesizing delay-resilient control.* Acta Informatica '20.