Planning using UPPAAL SMC & UPPAAL STRATEGO

Peter G. Jensen, May the Fourth, 2021

AALBORG UNIVERSITET

Overview

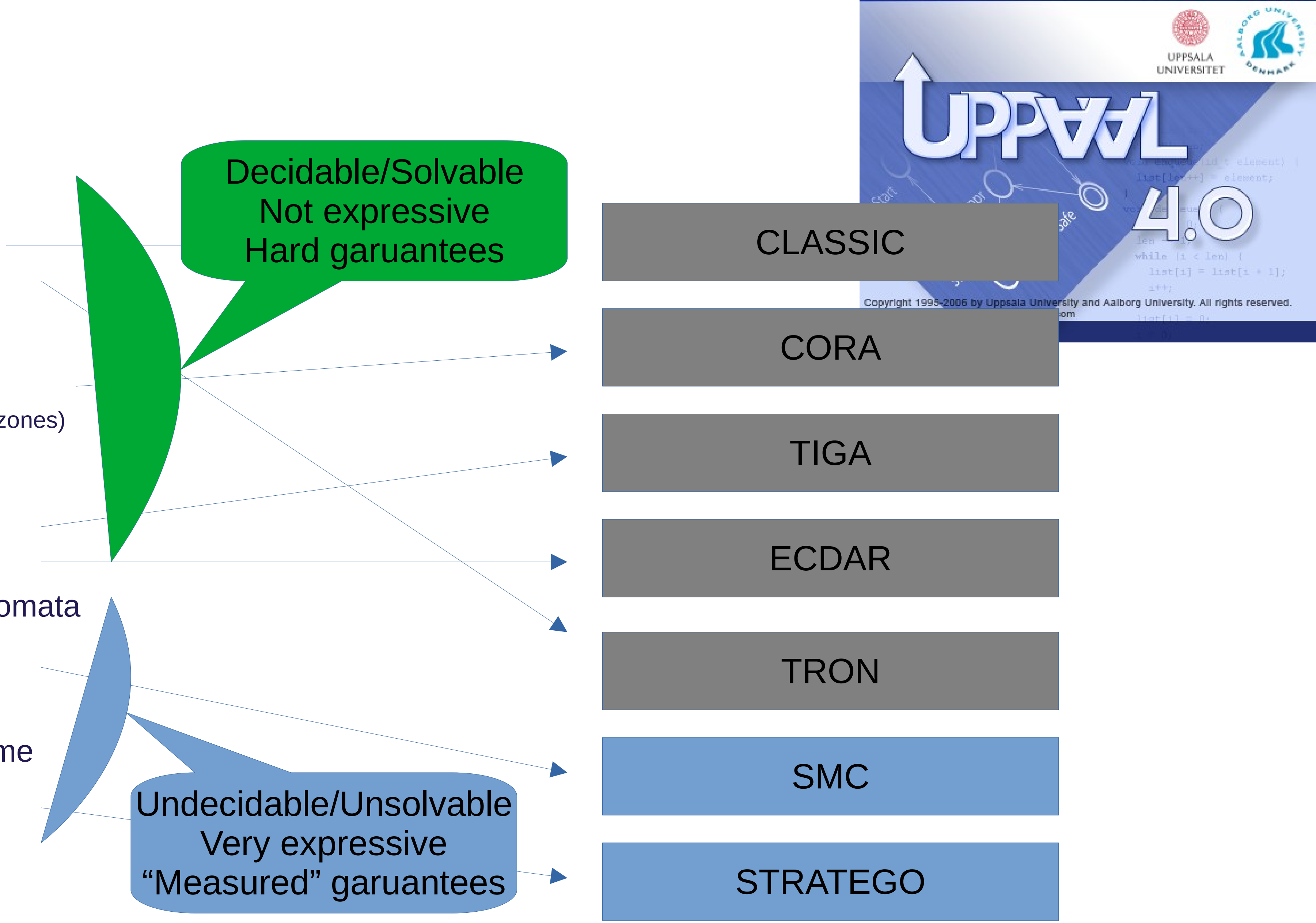
Timed Automata
- Decidability (regions)
- Symbolic Verification (zones)

Priced Timed Automata
- Decidability (priced regions)
- Symbolic Verification (priced zones)

Timed Games
- Decidability (regions)
- Symbolic Verification (zones)

Stochastic Priced Timed Automata
- Stochastic semantics
- Statistical methods

Stochastic Priced Timed Game
- Stochastic Semantics
- Reinforcement Learning



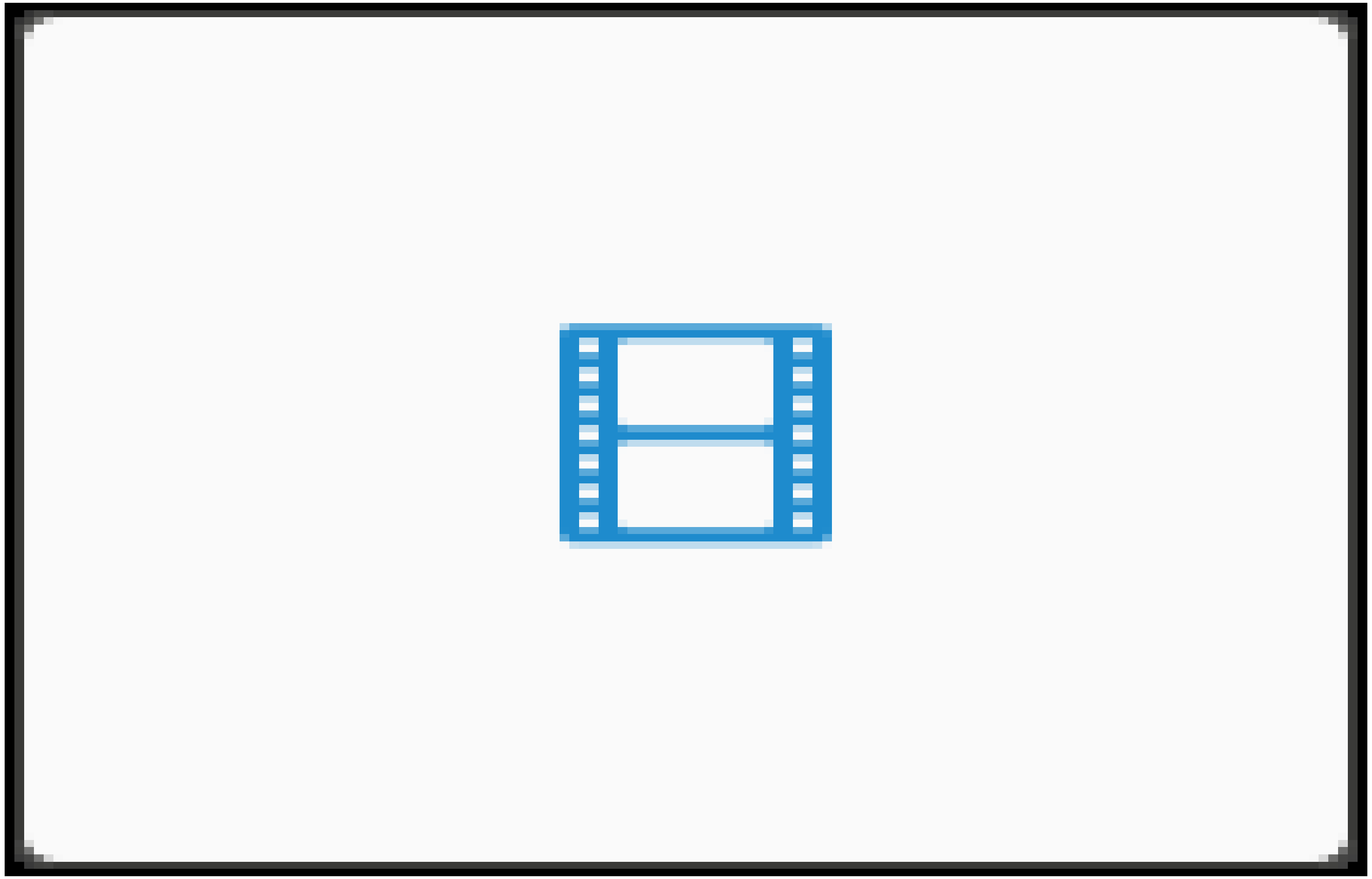
Outline

Stochastic Timed Automata (SMC)

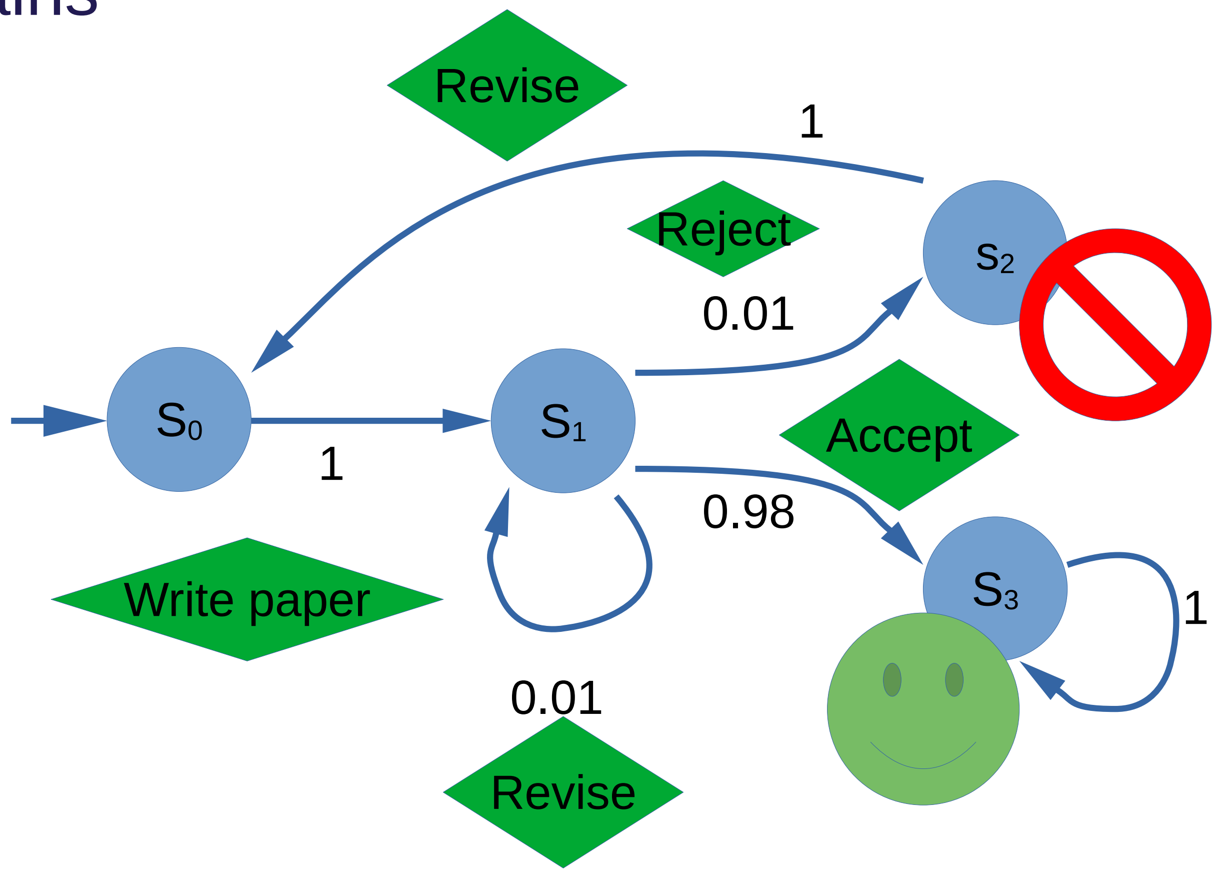
- Markov Chains
- Stochastics and Time
- Common Modeling Mistakes
- Exercise!
- Examples

Stochastic Priced Timed Games (Stratego)

- Markov Decision Processes
- Q-learning (Tabulation)
- Euclidean Markov Decision Processes
- Stratego
  - Safe learning!
  - Partial observability
- Exercise!
- Examples

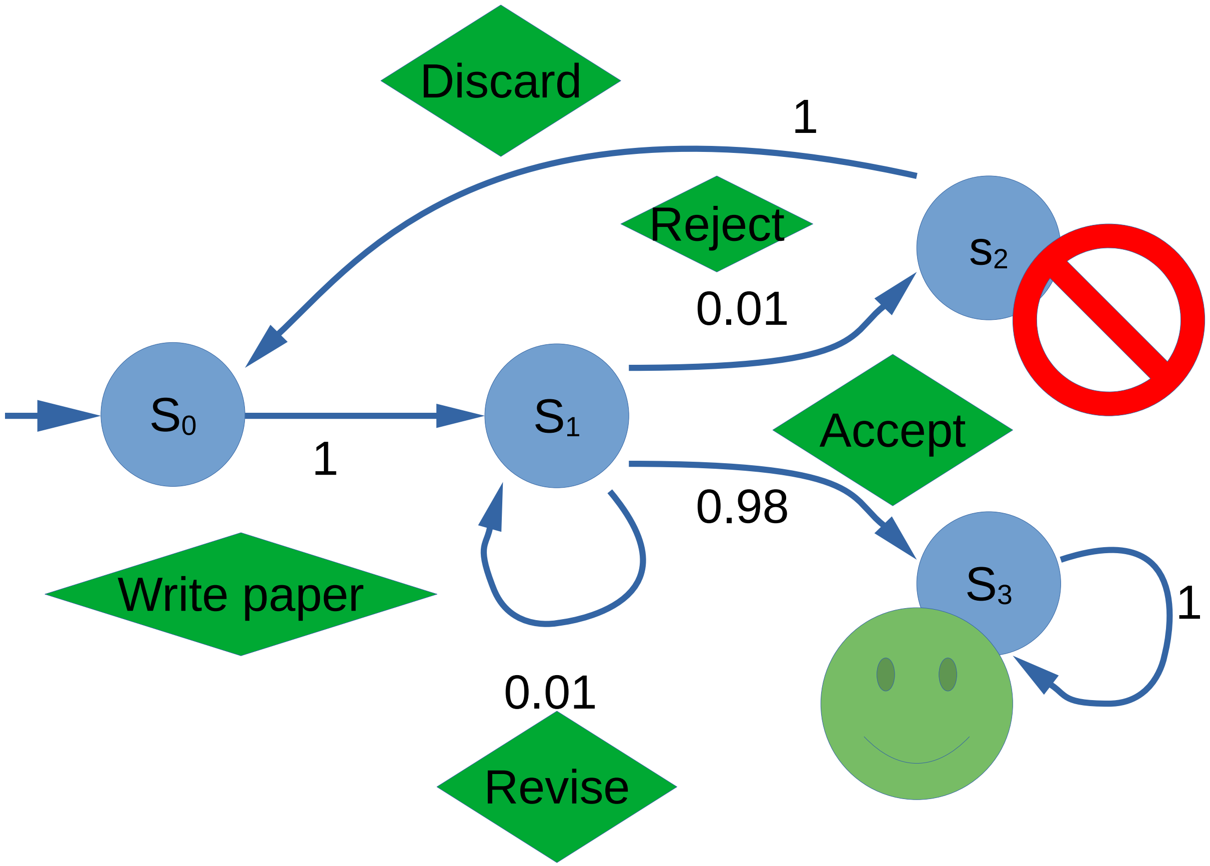


Stochastic Semantics
Markov Chains



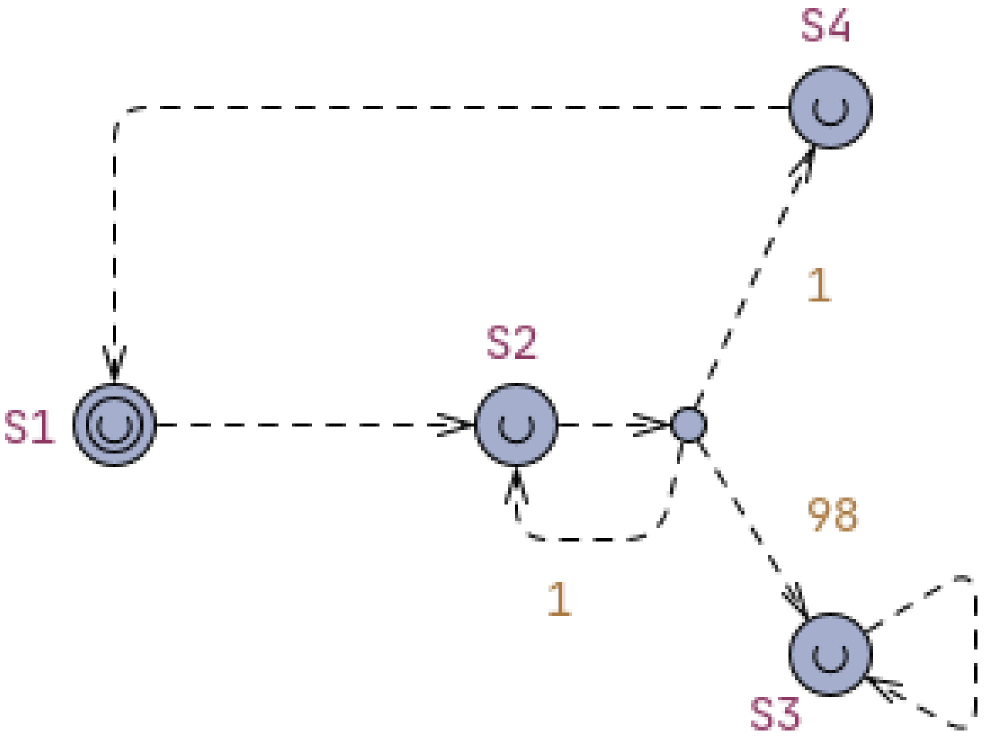
$Pr(\,[\,]\,\neg S_2) = 0.98/0.99 \approx 0.98989898...$

Stochastic Semantics
Markov Chains



Pr([ ] ¬ S_2) = 0.98/0.99 ≈ 0.98989898...

… In UPPAAL!



Notice, no time yet, all locations are urgent!

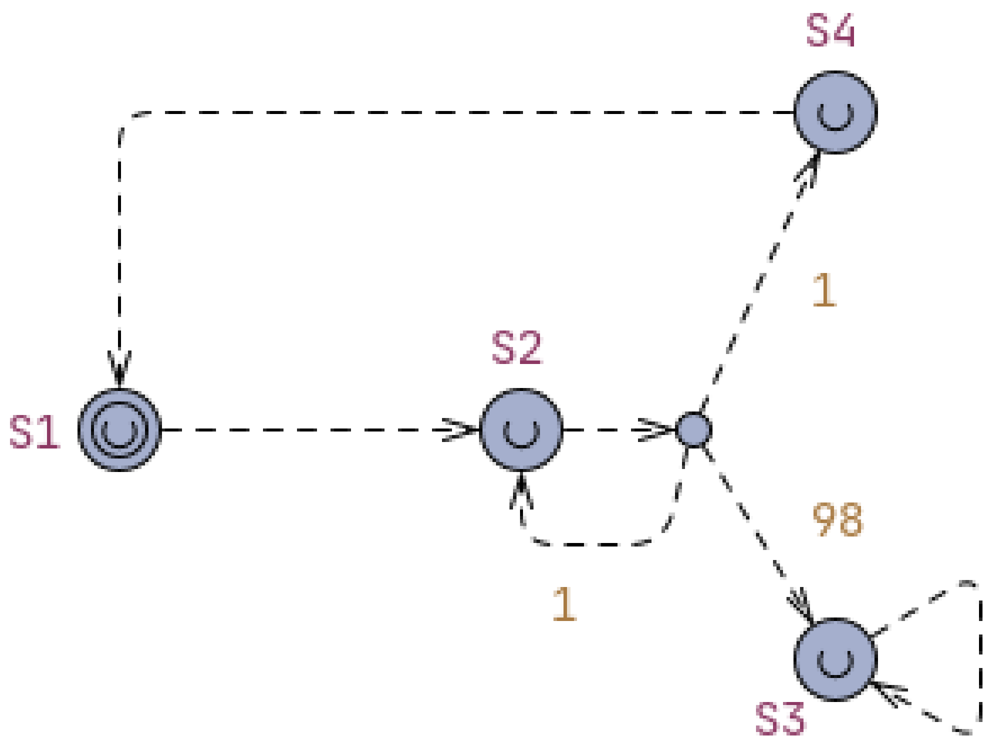
Stochastic Semantics
vs Nondeterministic Semantics

We can loop S1,S2* forever

Nondeterministic:
A<> S3

Will we always reach S3?

Stochastic/probabilistic:
Pr(<> S3)



Probability tends to 1 over infinite execution

Stochastic Semantics
vs Nondeterministic Semantics

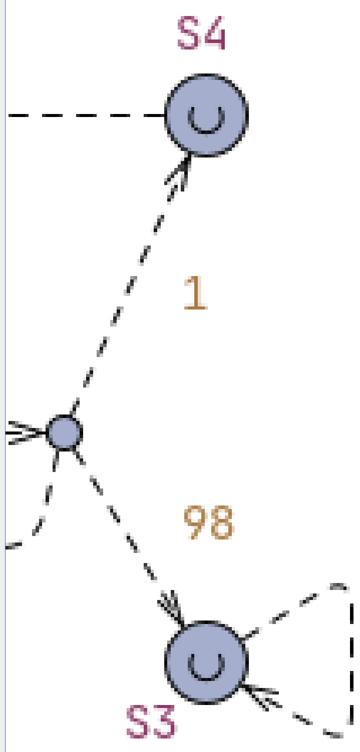
We can loop

Nondet
A<

Will we alwa

Stochastic/
Pr(<> S3)

Lesson:
[Pr(X) = 1] ≠ [A<> X]



Probability tends to 1 over infinite execution

Mini Exercise

You want to play a game of dices – but you only have a coin avalible (heads/tails).
Implement an emulator of a 6-dice using coinflips.

Ensure that the dice is fair!
(What is a good specification?)

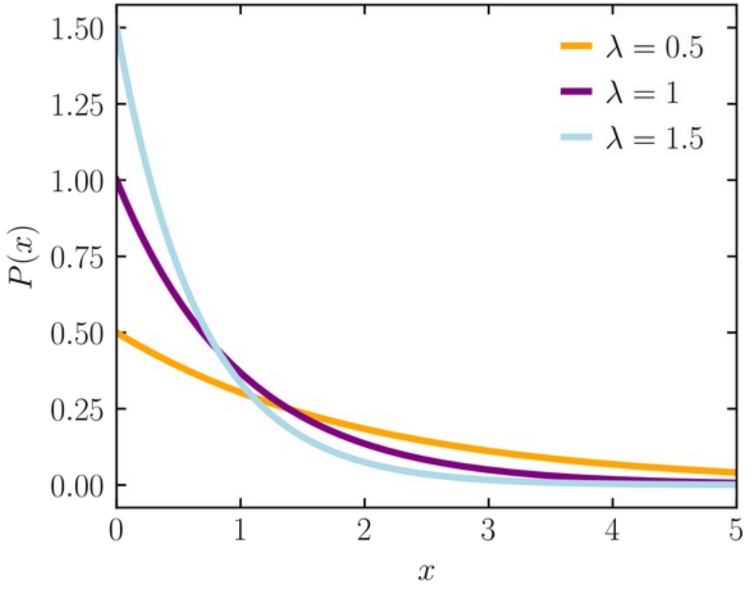
You can use

```
Pr[<=1](<> Process.L)
```

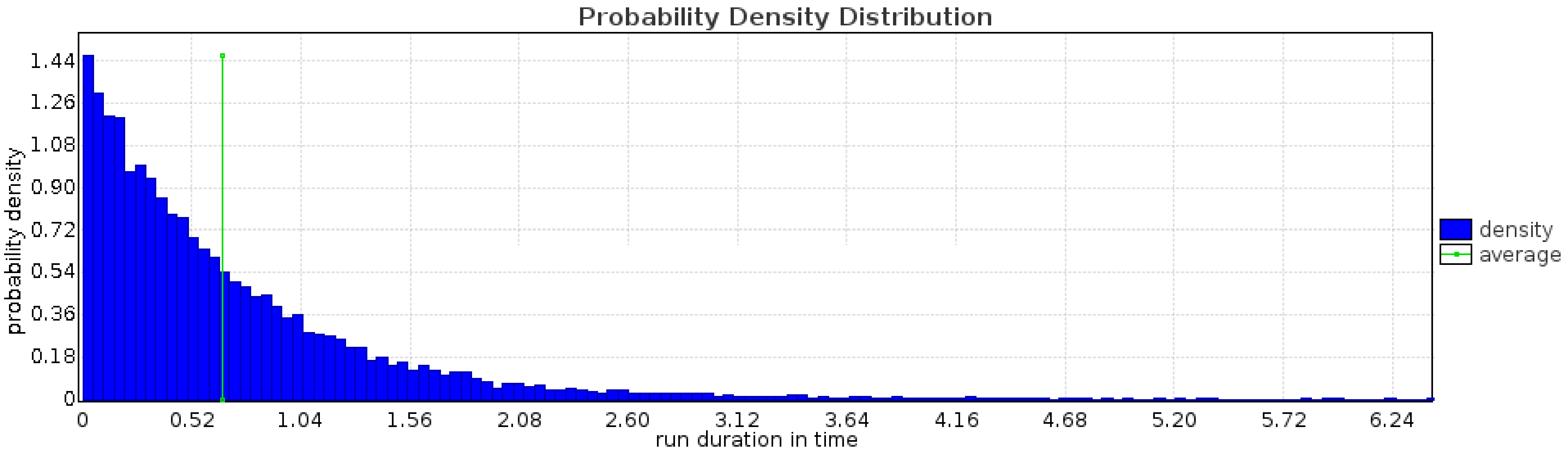
to estimate the probability of ending in a location L of Process

Tip:
Make all (but terminal) locations urgent!

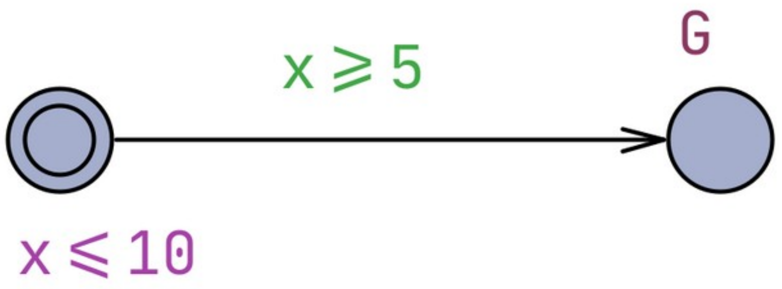
Stochastic Semantics and Time

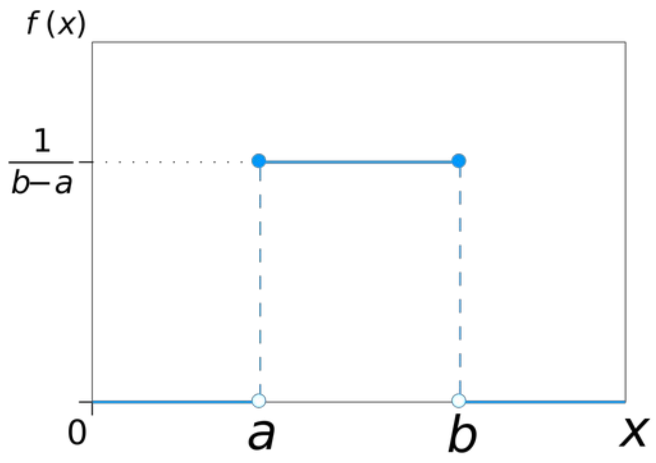


Pr(<> G)=

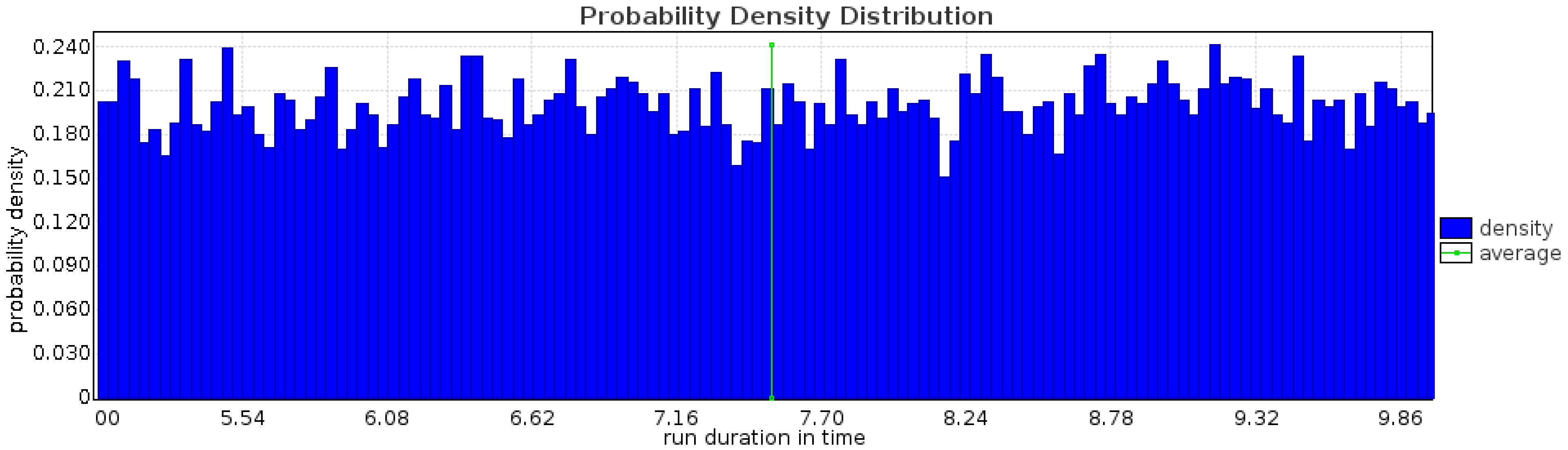


Stochastic Semantics and Time

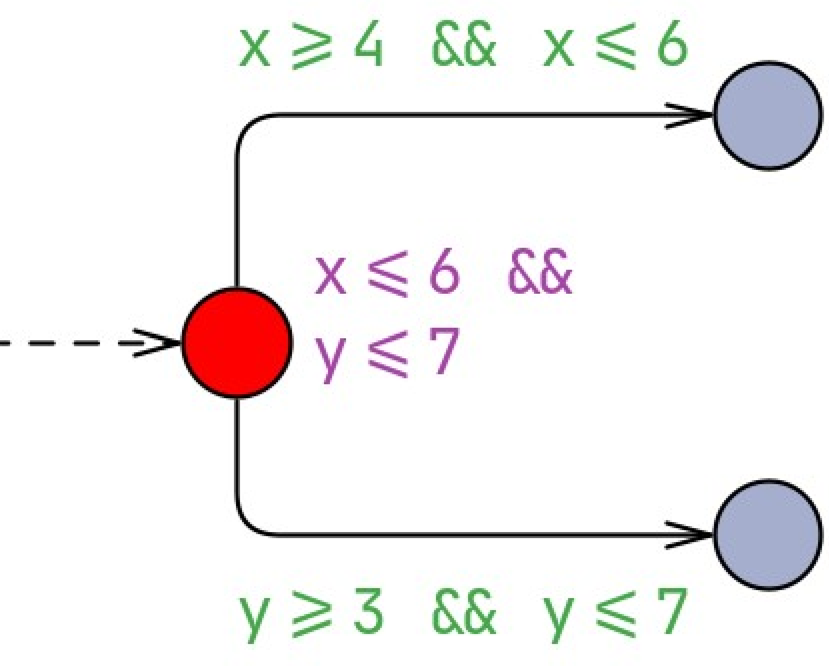


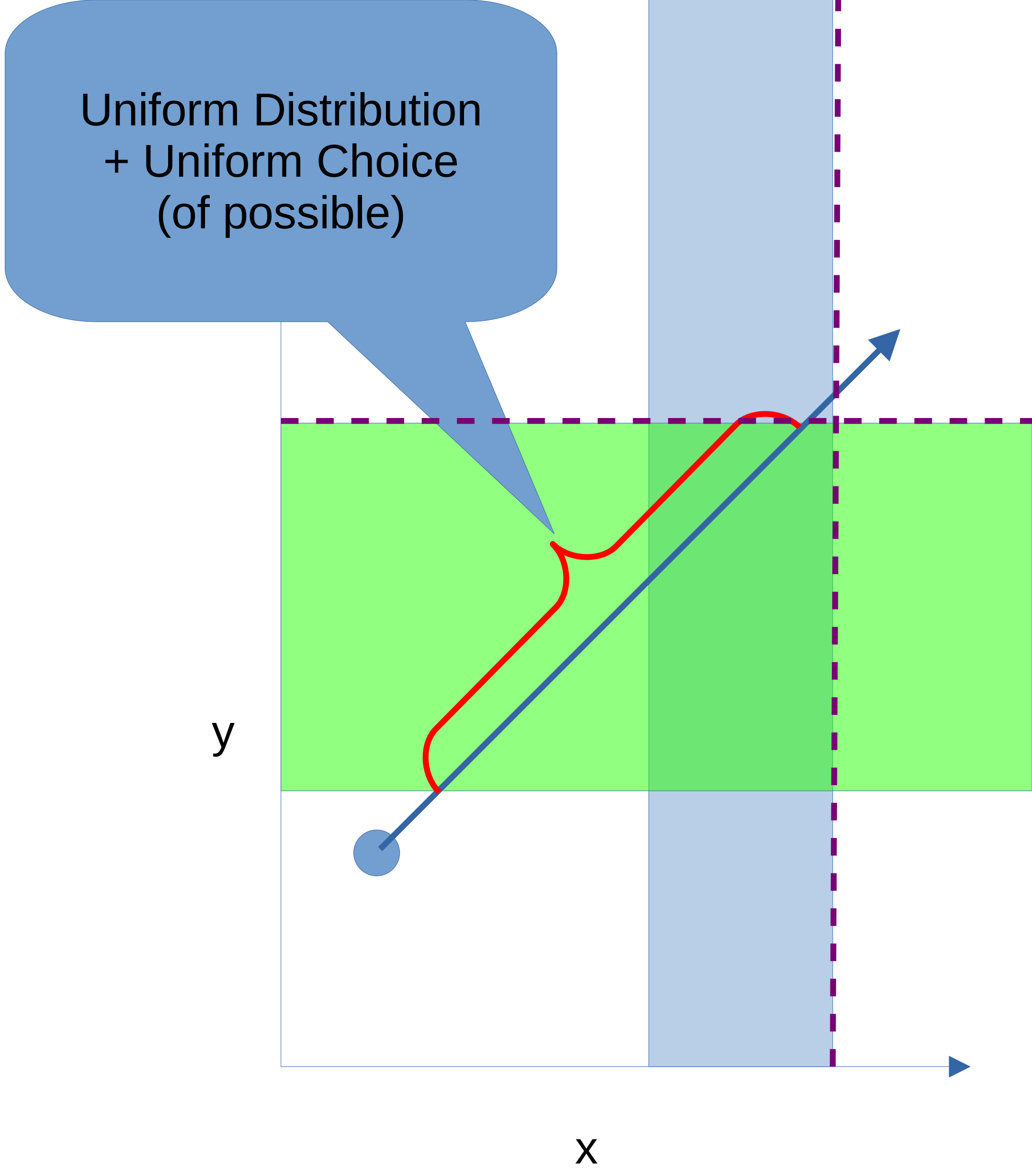


Pr(<> G)=

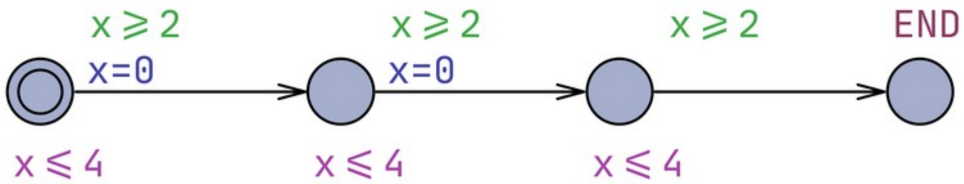


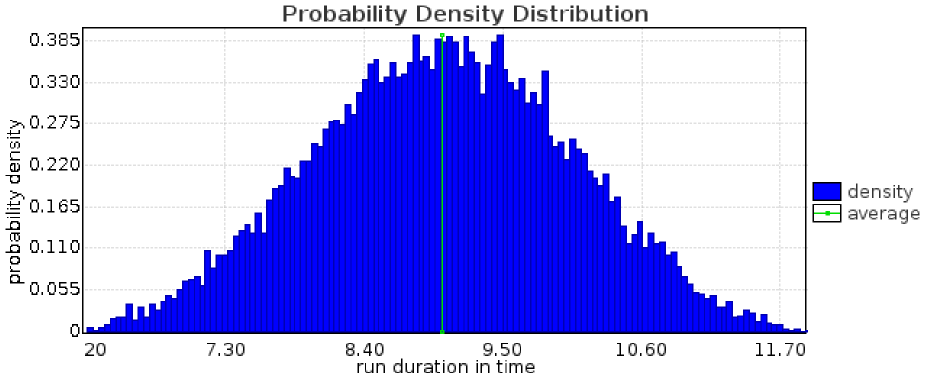
Stochastic Semantics
and Multiple Temporal Choices

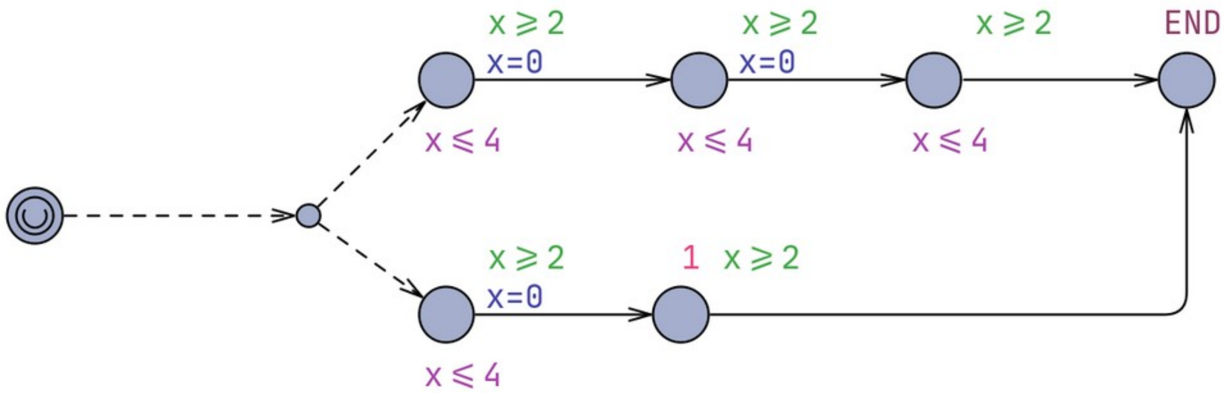


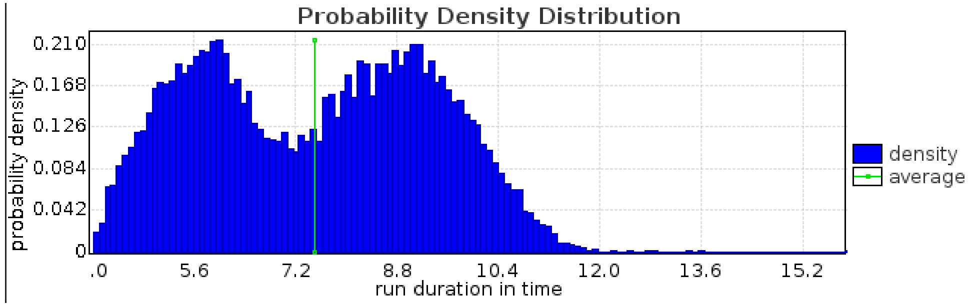


Stochastic Semantics and Phase-Type Distributions





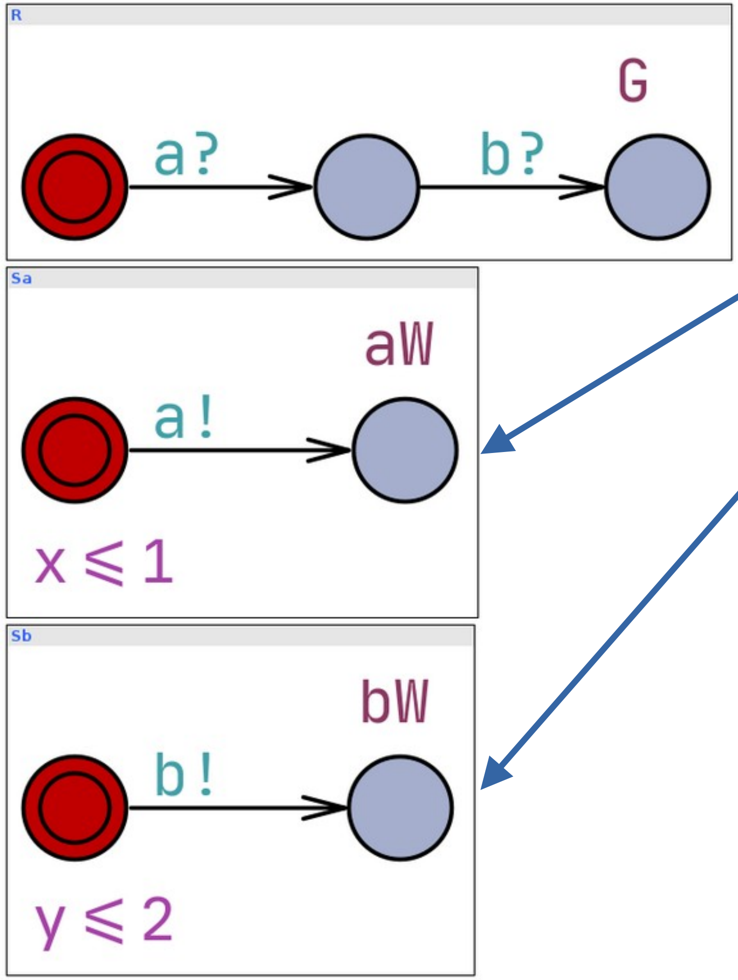




Any distribution can be encoded to arbitrary precision

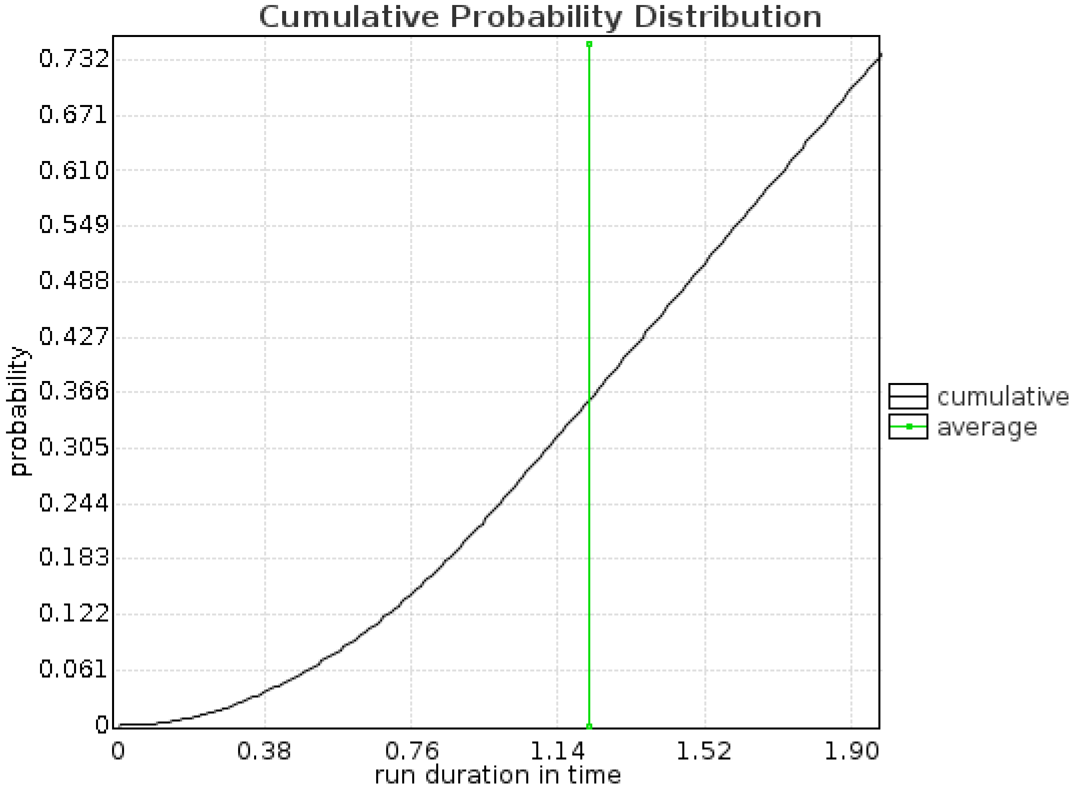
Calls to external C-function allow for arbitrary distributions

Stochastic Semantics
and Multiple Automata



Race Between components

Pr(<> G)= 0.75



Queries of SMC

Notice:
Timebound is always needed!
It is a sampling-based method.

Evaluation

```
Pr[<=100] (<> expr)
Pr[<=100] ([] expr)
```

Hypothesis Testing

```
Pr[<=100] (<>  expr) >= 0.1
Pr[<=100] ([]  expr) >= 0.1
```

Comparison

```
Pr[<=100] (<>  expr) >= Pr[<=10] (<>  expr2)
```

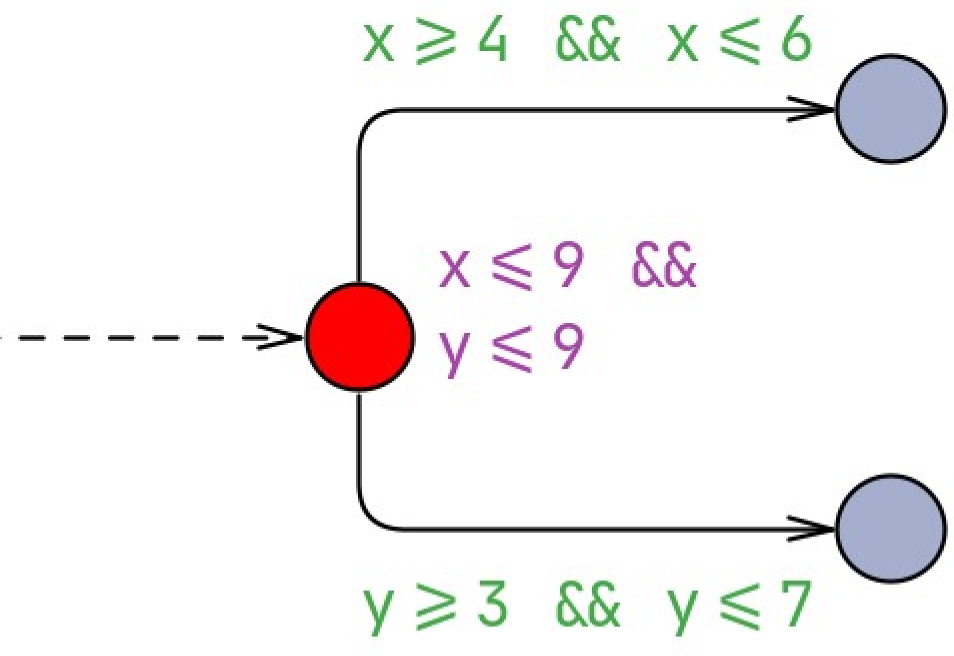
Expected Value

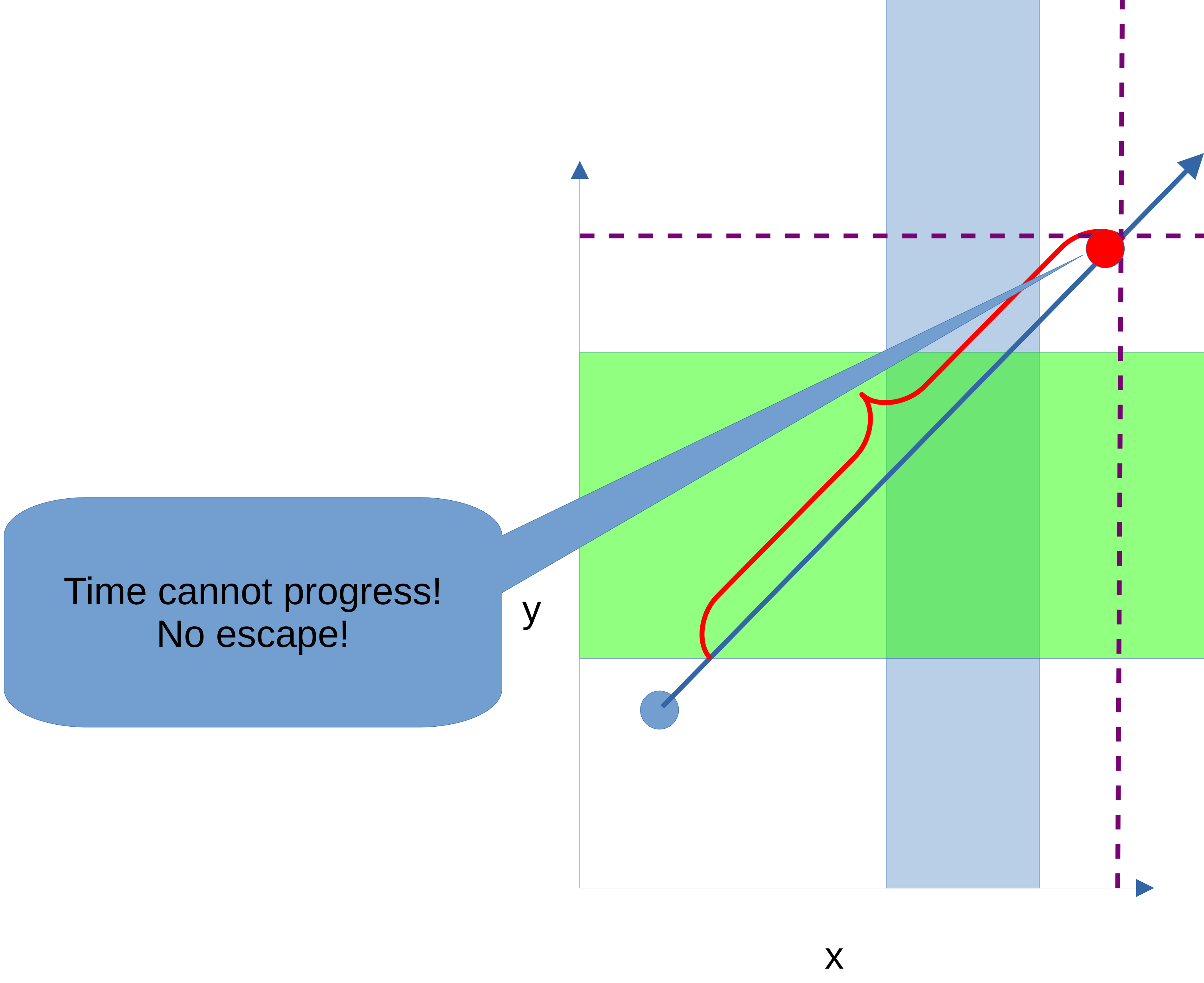
```
E[<=100;100] (min: expr)
E[<=100;100] (max: expr)
```

Simulation

```
simulate 1 [<=100] {e1, e2}
```

Common Modelling Mistakes
Time Locks





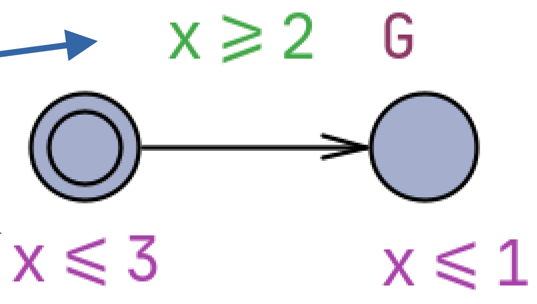
Common Modelling Mistakes
Model Sanity

Different to Model Checking (Classic)
We cannot impose constraints backwards

Delay must be
In [2,3]

$\Leftrightarrow$

x is in [2,3] after delay



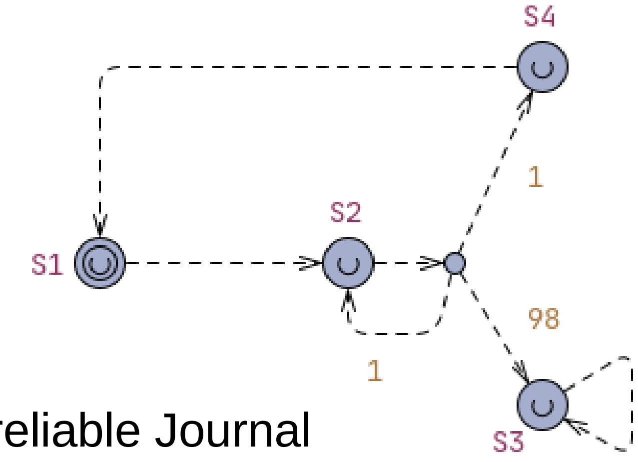
Valid successor must have
$x \leq 1$

Simmulation allows for constructions that make such analysis/imposition theoretically impossible

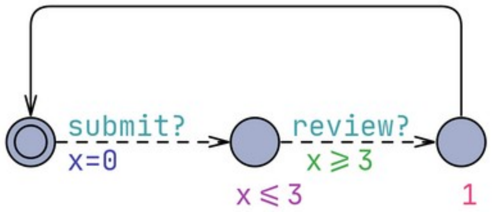
Exercises

1) Temporal Papers

- Extend the paper-writing model s.t. the researcher starts writing a paper uniformly between 1 and 6 months.
- It takes the researcher one month (with an exponential distribution) to revise the paper.
- If the paper is rejected, an exponential recuperating period is mandated (2 months).
- If the paper is accepted, the researcher starts over.
- What is the expected number of papers produced over a 5 year period?
- Connect three researchers with the “Unreliable Journal”, what is the throughput (you may ignore the review signal)?

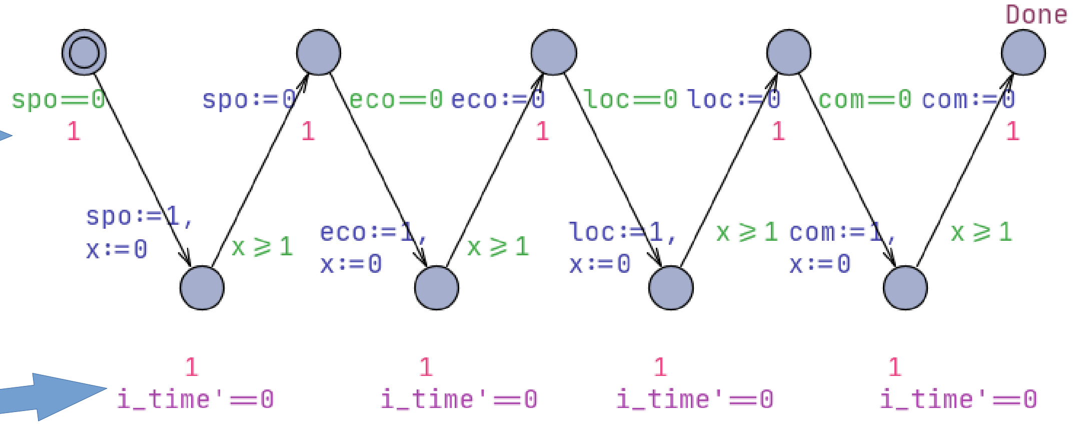


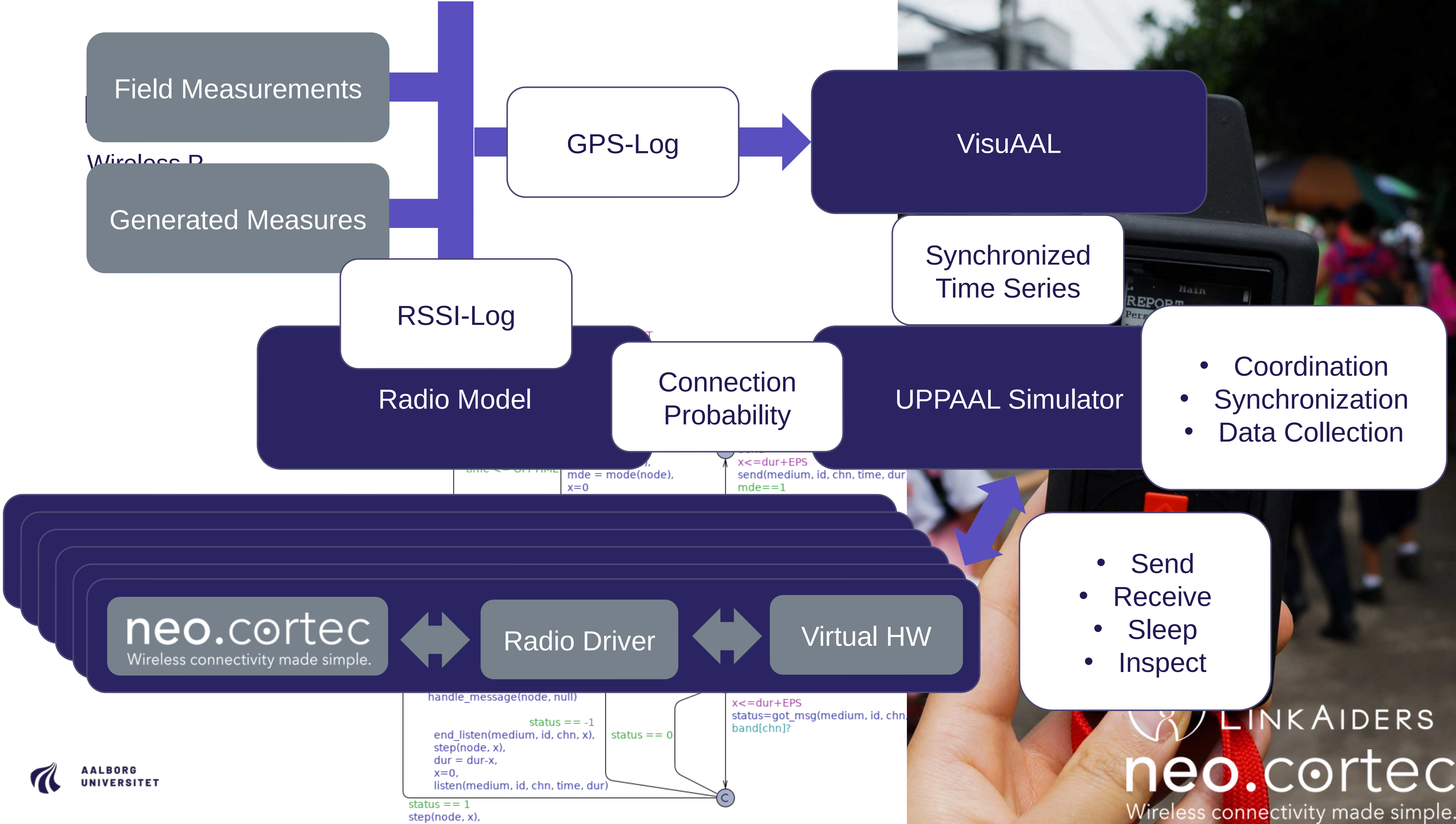
The Unreliable Journal



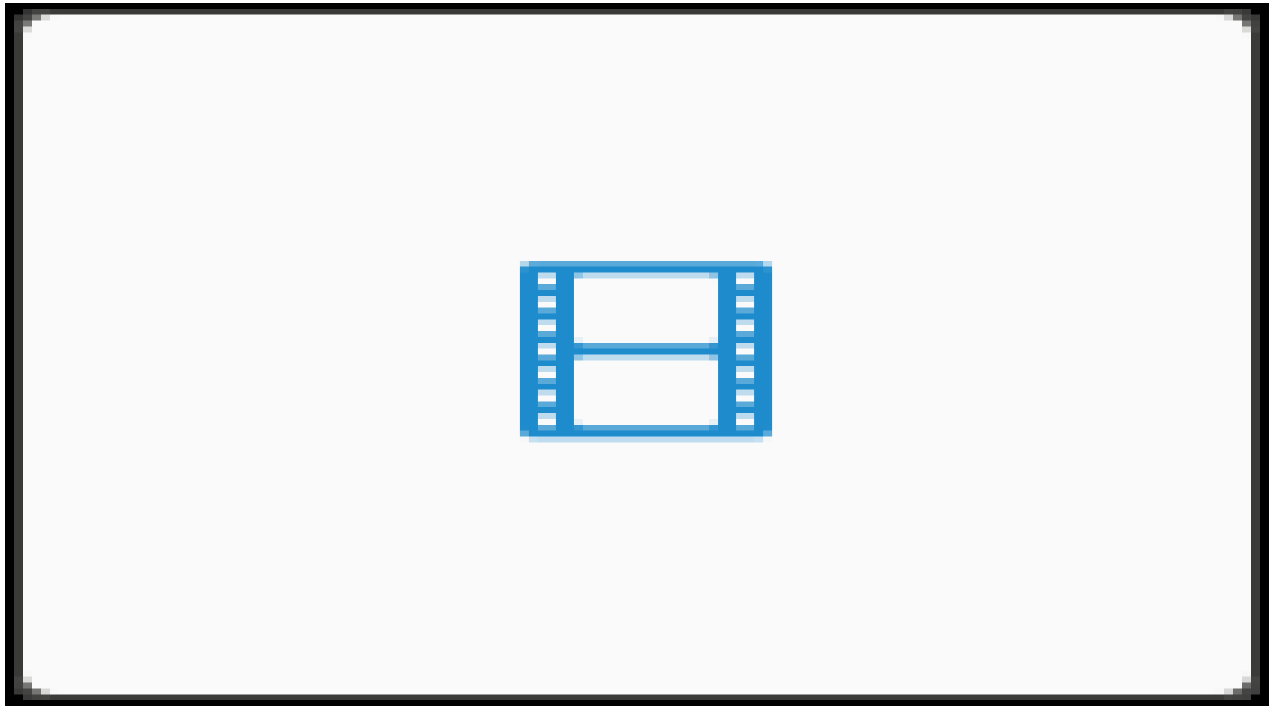
2) Stochastic Jobshop Scheduling

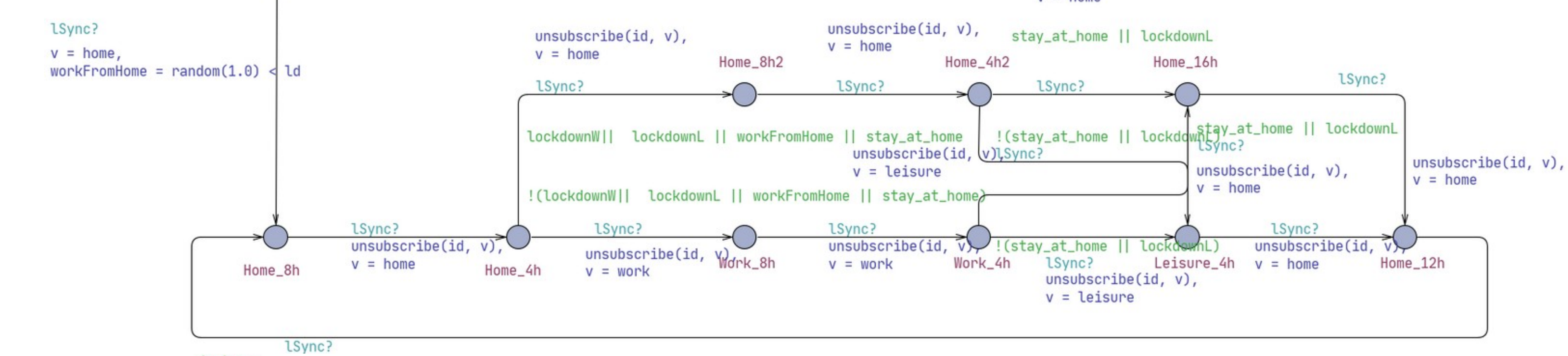
- Extend your Jobshop scheduling problem by adding distributions.
  - Assume the upperbound for reading is +5 timeunits.
  - Use a variable for the exponential rate for the “idle”-locations
- Study the effect of different exponential rates.
- Study the idle-duration of Kim
  - add a clock `i_time` with `i_time’==0` in the invariant in non-idle locations



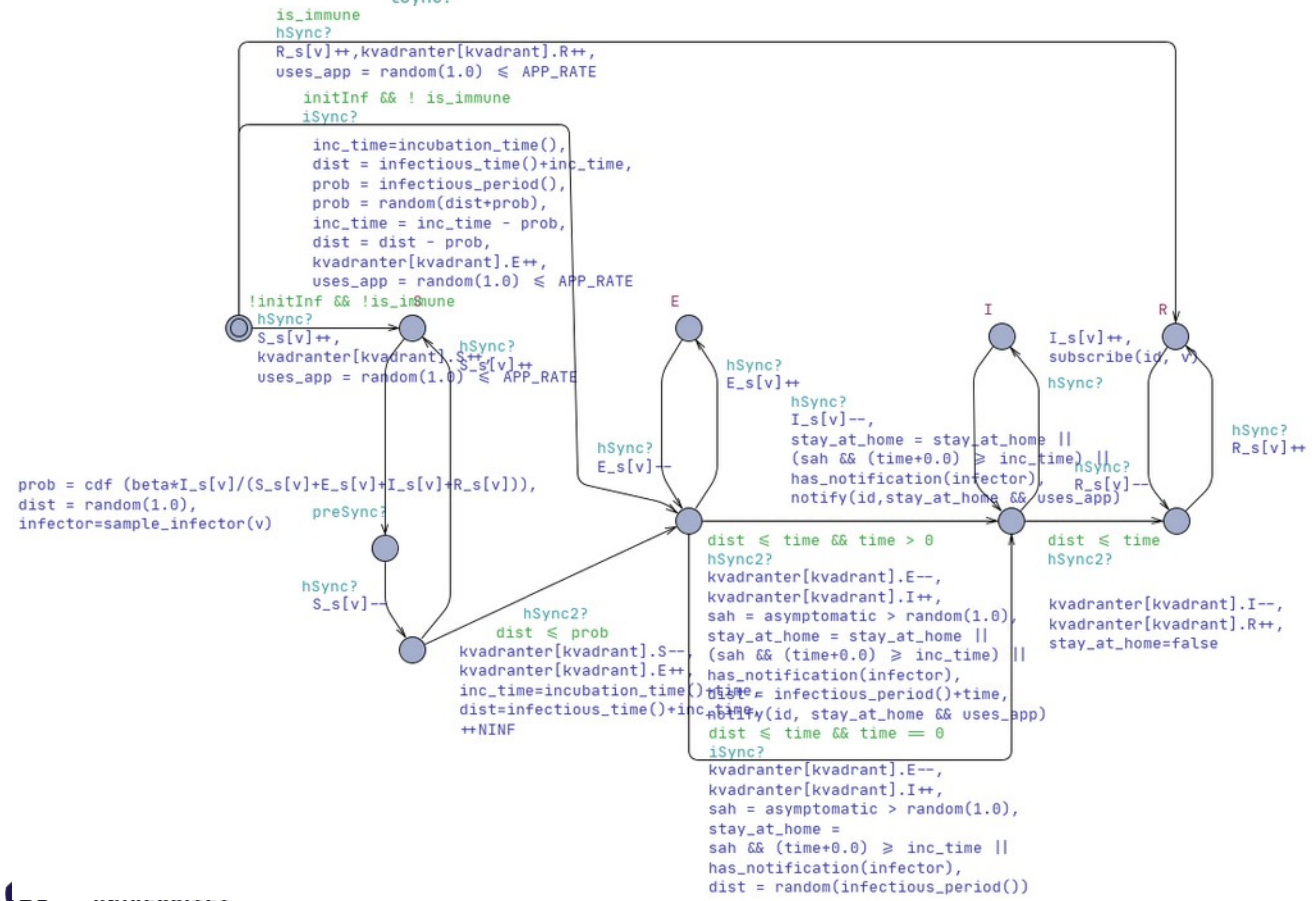


AALBORG UNIVERSITET

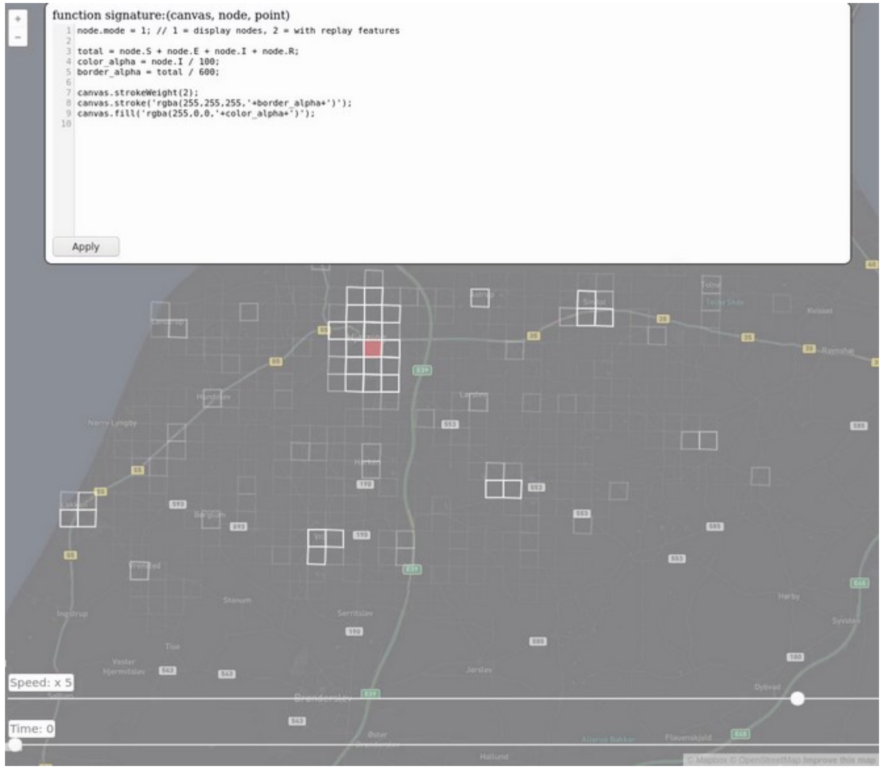




lSync?
v = home,
workFromHome = random(1.0) < ld
unsubscribe(id, v),
v = home
Home_8h2
unsubscribe(id, v),
v = home
stay_at_home || lockdownL
Home_4h2
Home_16h
lSync?
lockdownW|| lockdownL || workFromHome || stay_at_home
!(stay_at_home || lockdownL)
stay_at_home || lockdownL
unsubscribe(id, v),
v = leisure
unsubscribe(id, v),
v = home
!(lockdownW|| lockdownL || workFromHome || stay_at_home)
Home_8h
Home_4h
Work_8h
Work_4h
Leisure_4h
Home_12h
v = work
v = leisure



lSync?
is_immune
hSync?
R_s[v]++,kvadranter[kvadrant].R++,
uses_app = random(1.0) ⩽ APP_RATE
initInf && ! is_immune
iSync?
inc_time=incubation_time(),
dist = infectious_time()+inc_time,
prob = infectious_period(),
prob = random(dist+prob),
inc_time = inc_time - prob,
dist = dist - prob,
kvadranter[kvadrant].E++,
uses_app = random(1.0) ⩽ APP_RATE
!initInf && !is_immune
S
E
I
R
S_s[v]++,
kvadranter[kvadrant].S++,
uses_app = random(1.0) ⩽ APP_RATE
E_s[v]++
I_s[v]--,
stay_at_home = stay_at_home ||
(sah && (time+0.0) ⩾ inc_time) ||
has_notification(infector),
notify(id,stay_at_home && uses_app)
I_s[v]++,
subscribe(id, v)
R_s[v]--
R_s[v]++
E_s[v]--
prob = cdf (beta*I_s[v]/(S_s[v]+E_s[v]+I_s[v]+R_s[v])),
dist = random(1.0),
infector=sample_infector(v)
preSync?
S_s[v]--
hSync2?
dist ⩽ prob
kvadranter[kvadrant].S--,
kvadranter[kvadrant].E++,
inc_time=incubation_time()+time,
dist=infectious_time()+inc_time,
++NINF
dist ⩽ time && time > 0
kvadranter[kvadrant].E--,
kvadranter[kvadrant].I++,
sah = asymptomatic > random(1.0),
stay_at_home = stay_at_home ||
(sah && (time+0.0) ⩾ inc_time) ||
has_notification(infector),
dist = infectious_period()+time,
notify(id, stay_at_home && uses_app)
dist ⩽ time && time == 0
dist = random(infectious_period())
dist ⩽ time
kvadranter[kvadrant].I--,
kvadranter[kvadrant].R++,
stay_at_home=false

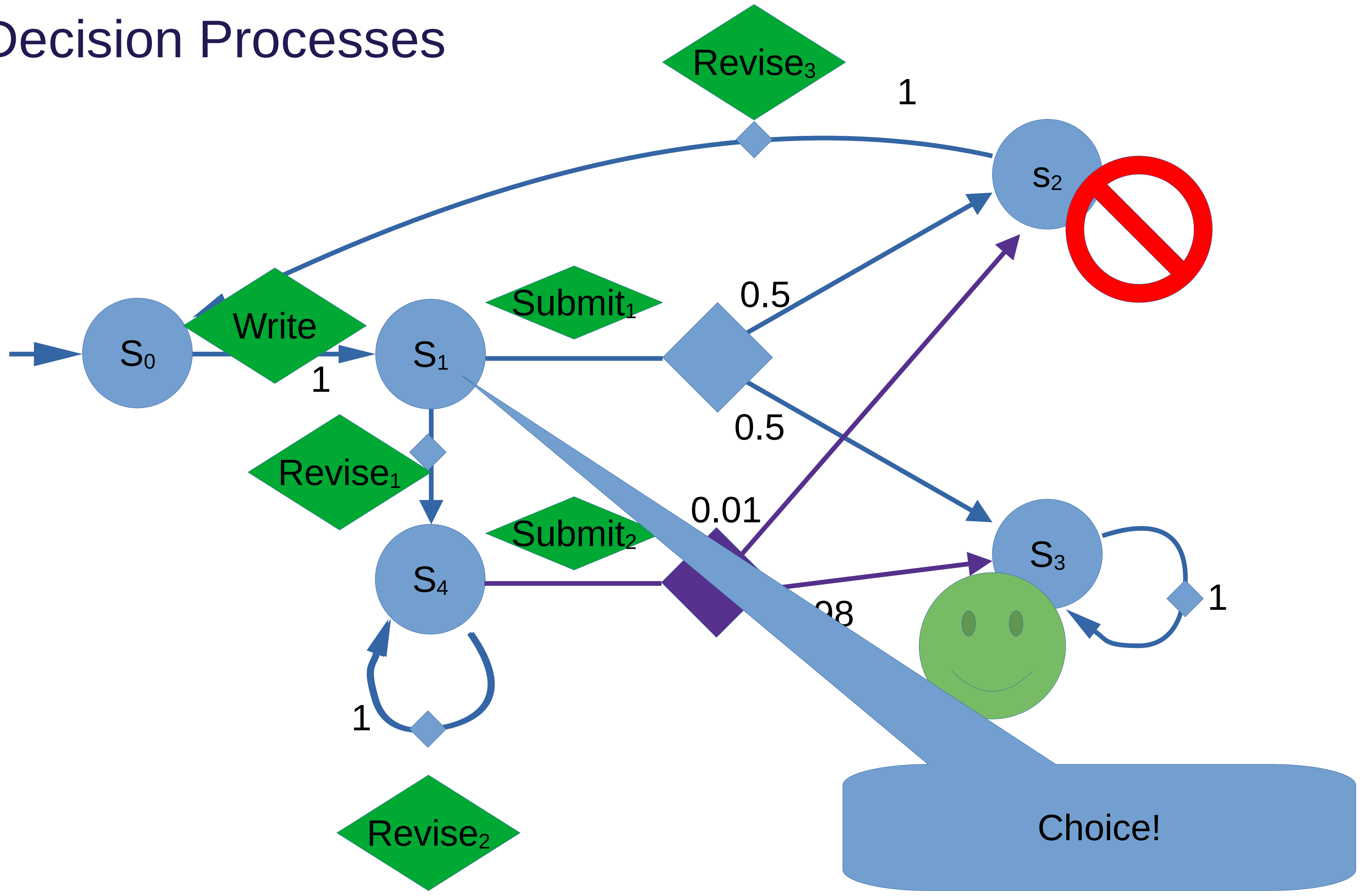


function signature:(canvas, node, point)
node.mode = 1; // 1 = display nodes, 2 = with replay features
total = node.S + node.E + node.I + node.R;
color_alpha = node.I / 100;
border_alpha = total / 600;
canvas.strokeWeight(2);
canvas.stroke('rgba(255,255,255,'+border_alpha+')');
canvas.fill('rgba(255,0,0,'+color_alpha+')');
Apply
Speed: x 5
Time: 0

Stochastic Optimization of Stochastic Priced Timed Games using Uppaal Stratego

AALBORG UNIVERSITET

Markov Decision Processes



Markov Decision Processes

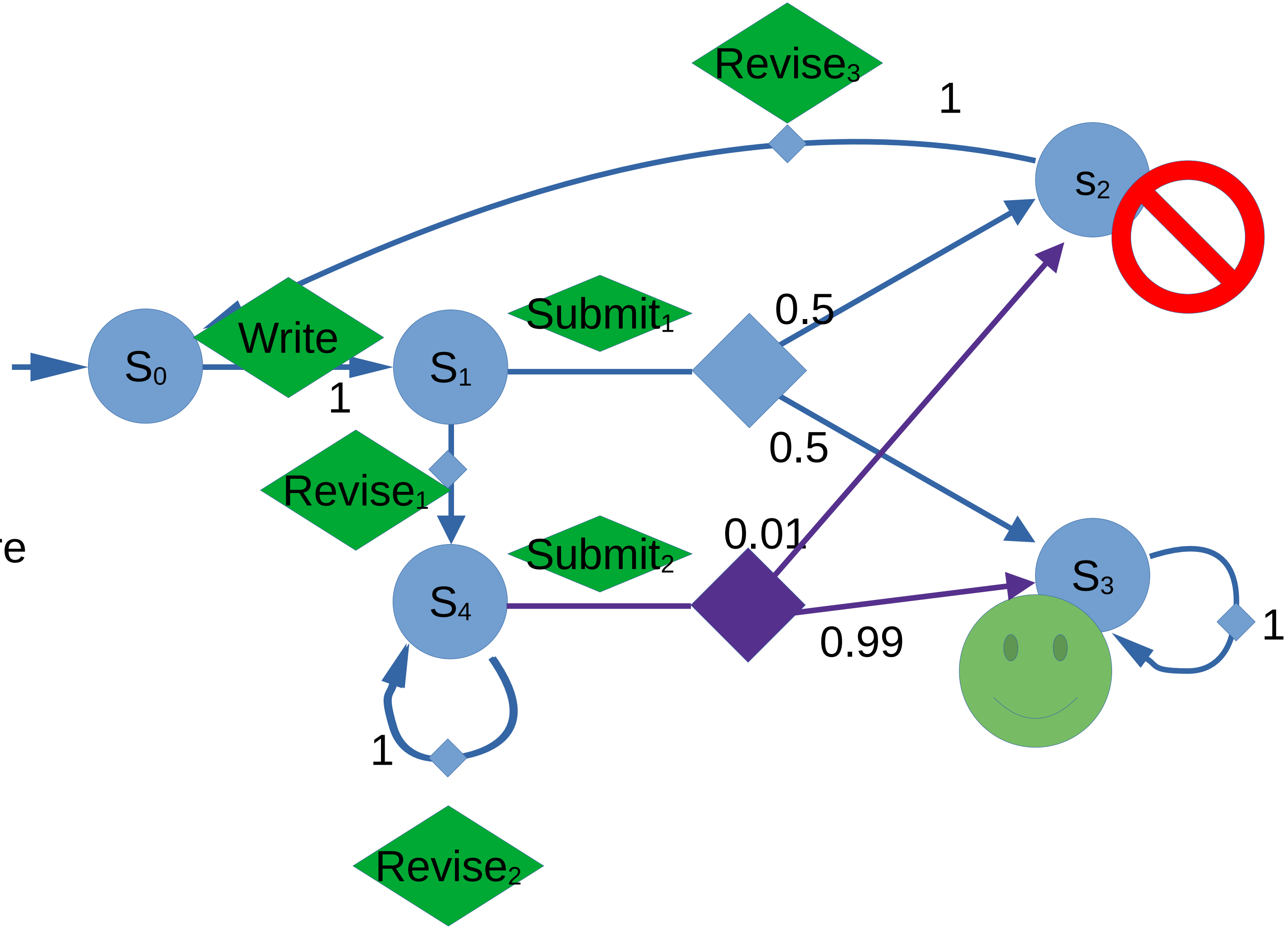
We could have costs on the actions

What is the optimal ***Strategy*** s.t. we publish a paper?

Strategy:

σ : Q -> Dist{Write, Submit$_x$, Revise$_x$}

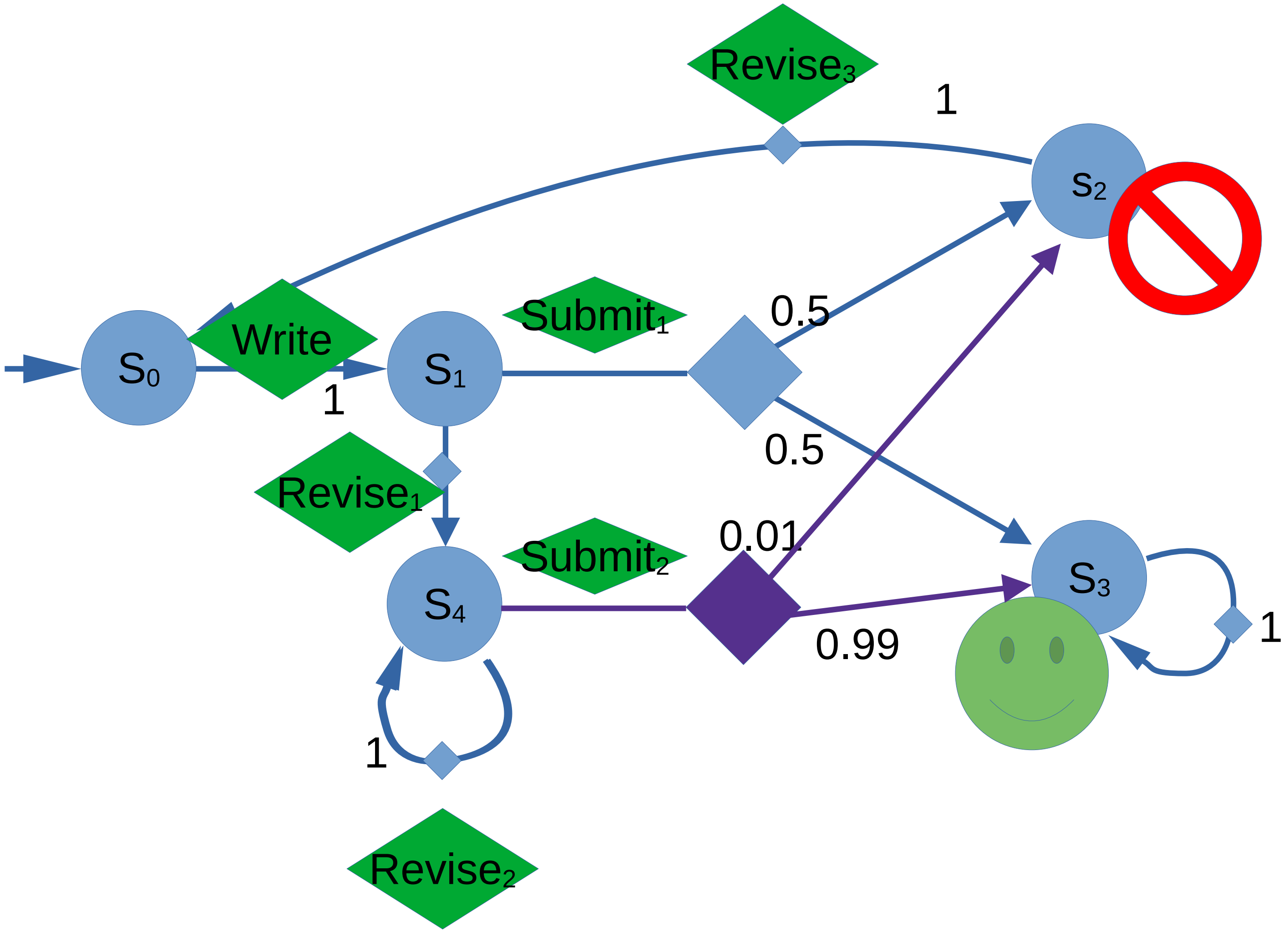
Theorem:
The optimal strategy can be computed and is defined by the Bellman equations. Furthermore a deterministic optimal strategy exists.



Q-learning

Model-free learning

- Transition probabilities are unknown
- Costs are unknown
- States are visible
- Actions are visible
- Works on samples of the system



Q-learning

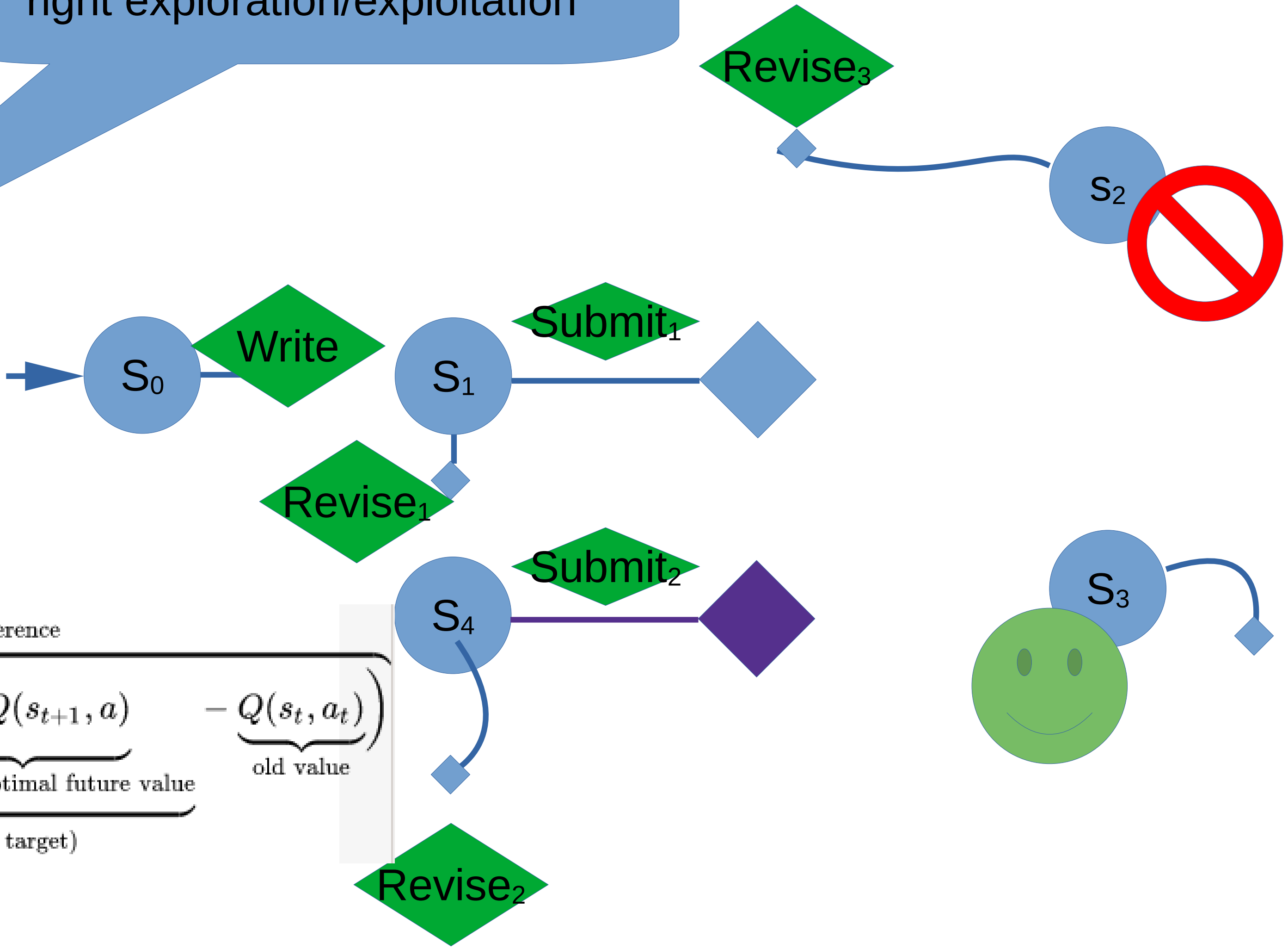
Model-free learning

- Transition probabilities are unknown
- Costs are unknown
- States are visible
- Actions are visible
- Works on samples of the system

In practice, pick the ‘right’ α and γ + right exploration/exploitation

Provable convergence to discounted optimum

… if the learning-rate converges
to zero in the limit AND
all actions have non-zero probability of
being sampled.



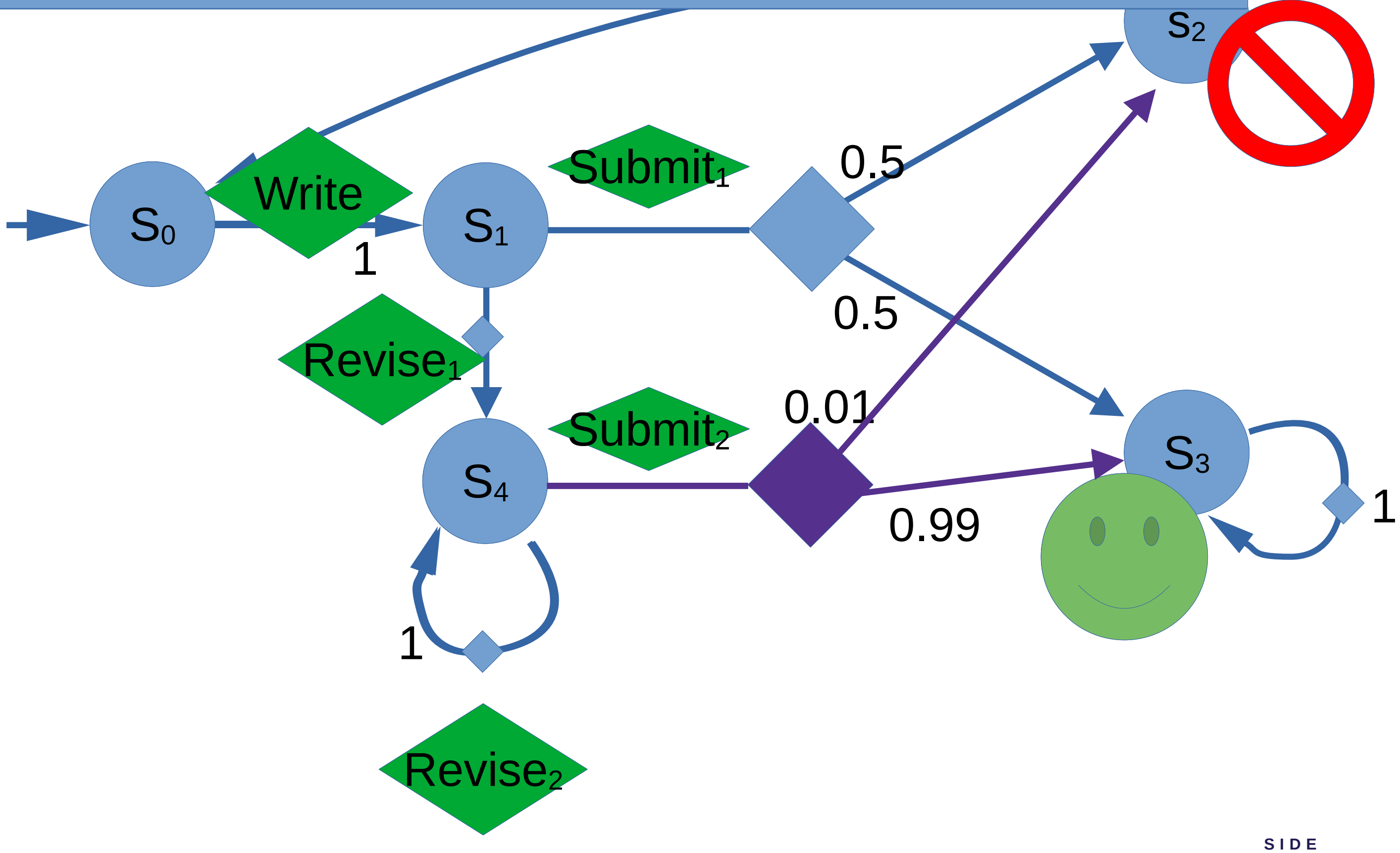
$$Q^{new}(s_t,a_t) \leftarrow \underbrace{Q(s_t,a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\bigg(\underbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1},a)}_{\text{estimate of optimal future value}}}_{\text{new value (temporal difference target)}} - \underbrace{Q(s_t,a_t)}_{\text{old value}}\bigg)}^{\text{temporal difference}}$$

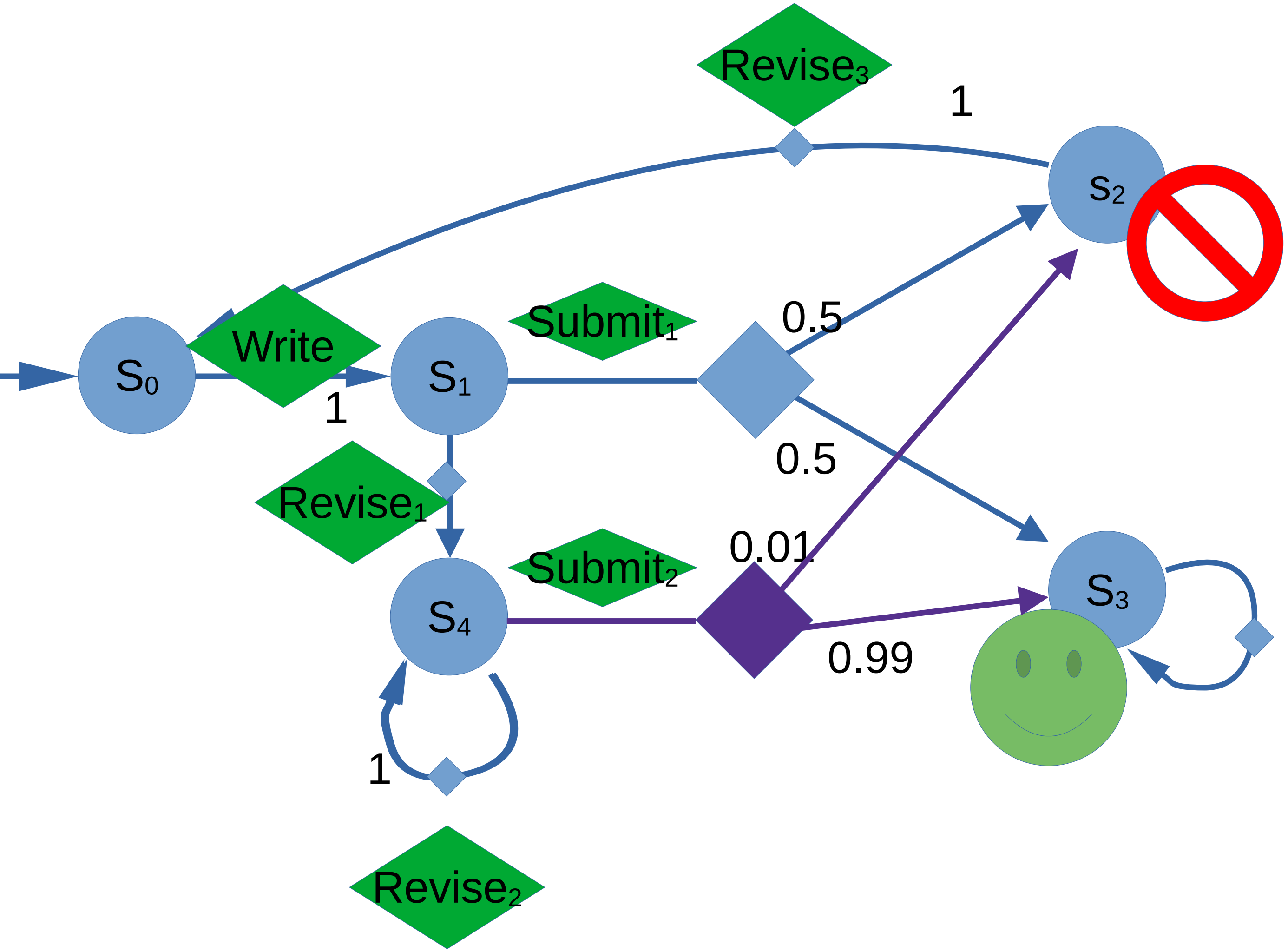
Assume action-reward -1, goal S_4,
Discount=0.9, learning-rate=0.5,
Greedy exploitation

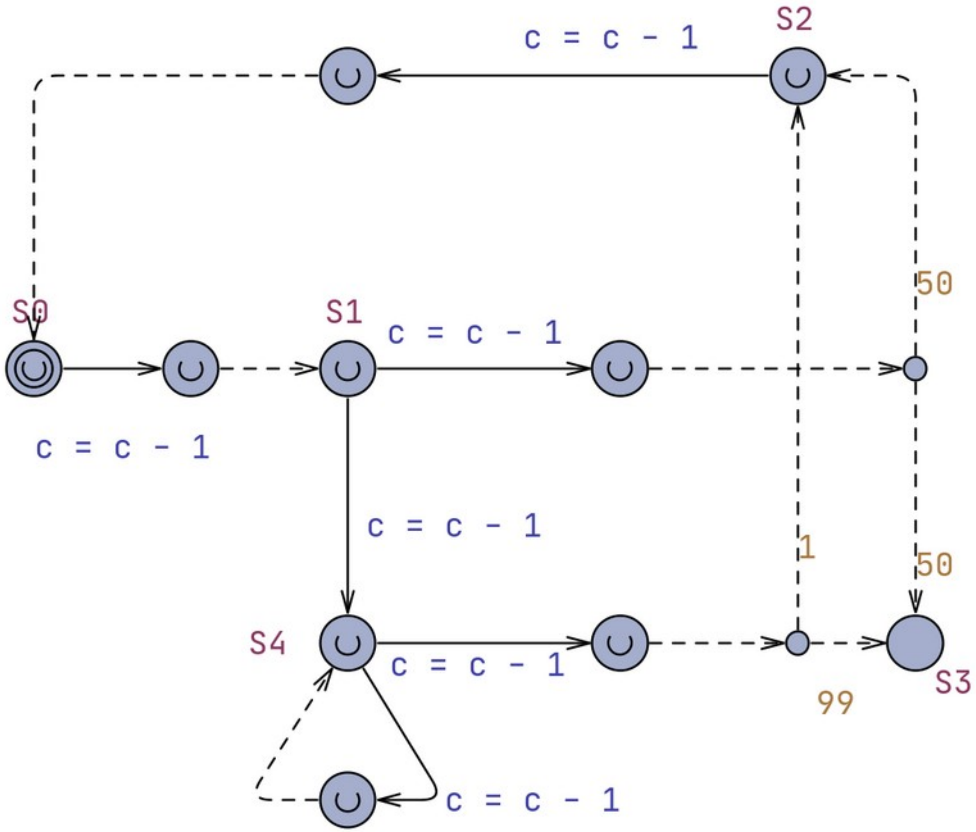
$$Q^{new}(s_t,a_t) \leftarrow \underbrace{Q(s_t,a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1},a)}_{\text{estimate of optimal future value}} - \underbrace{Q(s_t,a_t)}_{\text{old value}}\right)}^{\text{temporal difference}}$$

Q(S_n,	action)	=
S_0	Write	-0.75
S_1	$Submit_1$	-0.5
S_1	$Revise_1$	-0.75
S_2	$Revise_3$	0
S_3		0
S_4	$Submit_2$	-0.5
S_4	$Revise_2$	-0.75

.... lots of samples ...







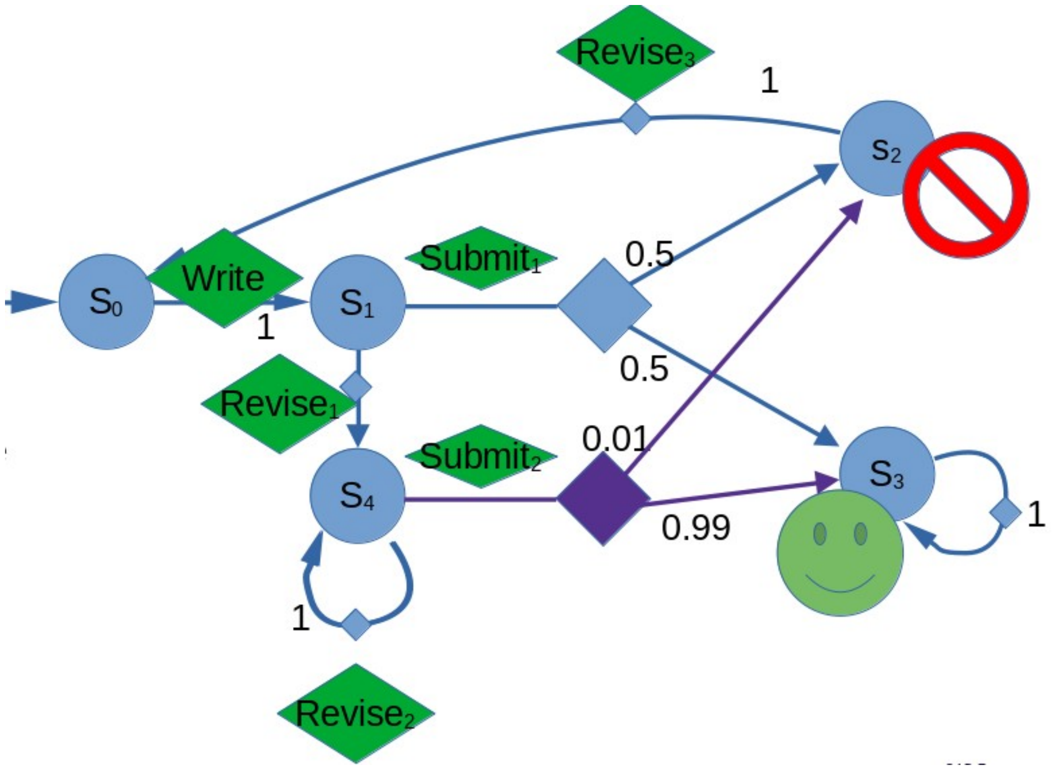

```
strategy s = maxE(c) [<=100]: <> S3  // compute the strategy
saveStrategy(s, "filepath.json")     // save the strategy
E[<=100;100] (max:c) under s         // evaluate strategy performance
E[<=100;100] (max:c)                 // performance of random controller
```

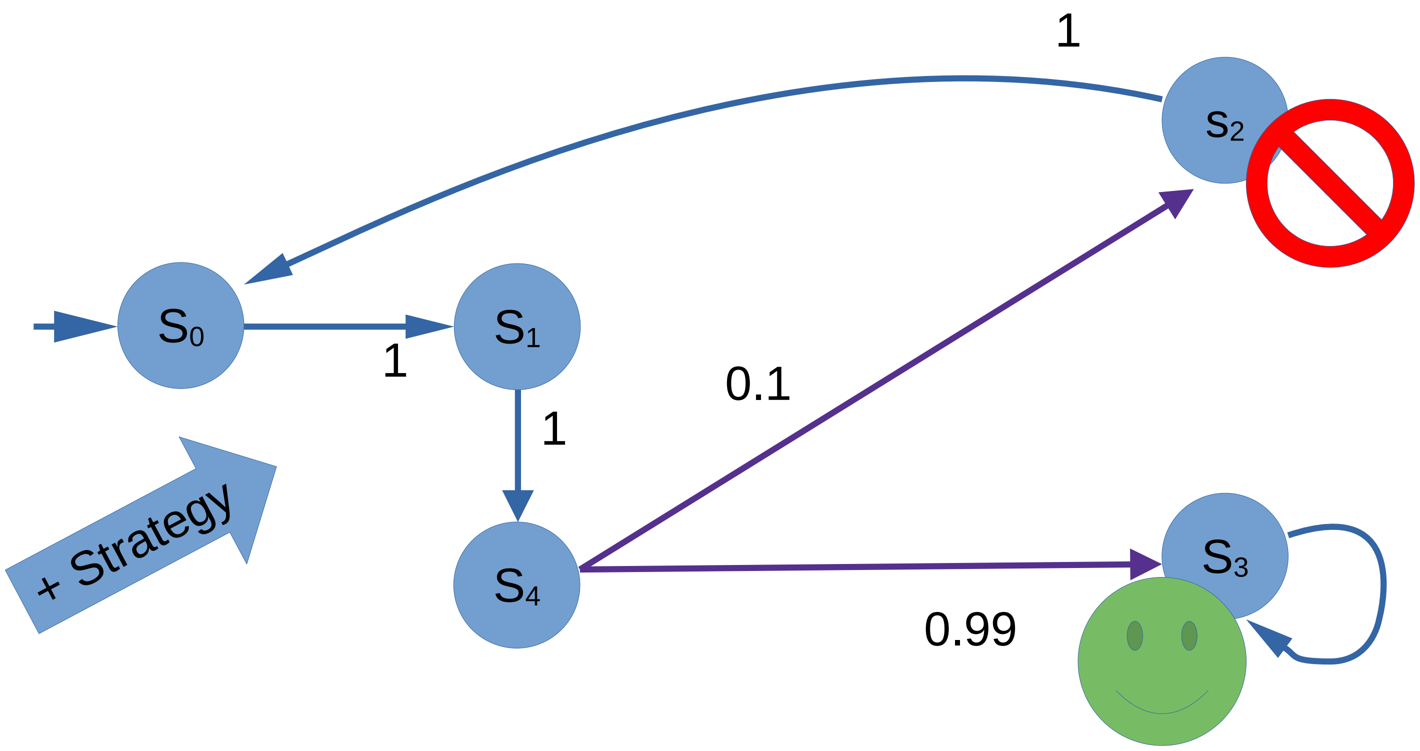
Q-Learning

Q-table gives us a stratgy

- For each state, pick “best” action

Strategy(Q-table) + MDP => MC

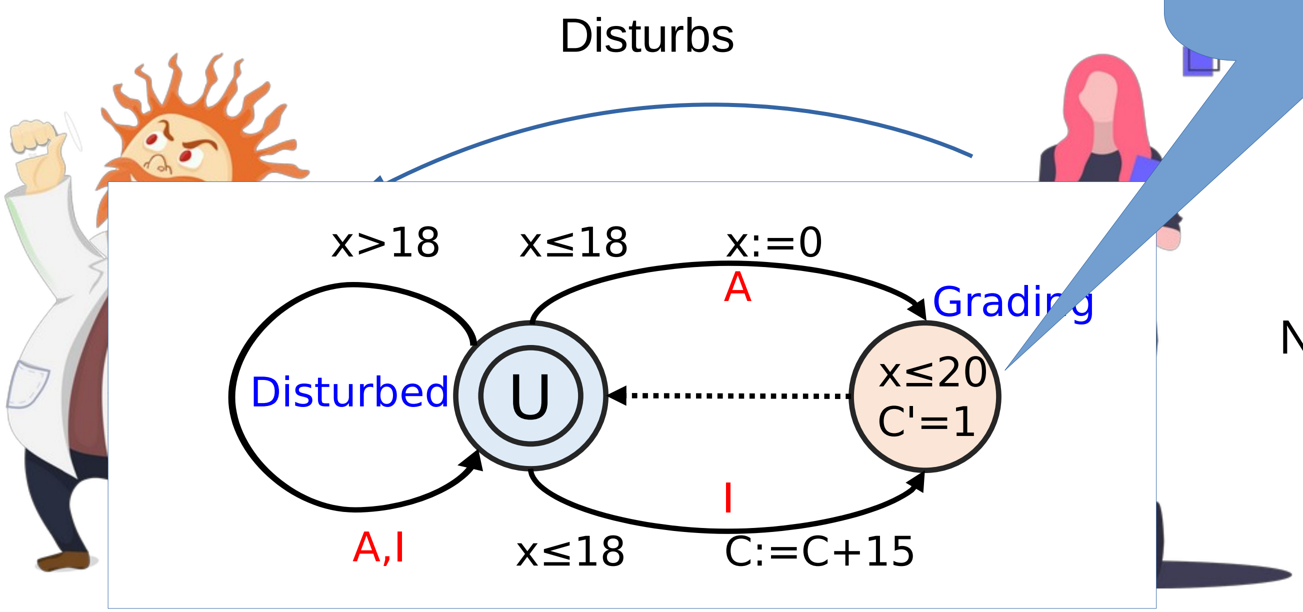




Euclidean Markov Decision Process
MDP With Continuous State Space

Uniform delay in [0,20]

Needs 18 hours for grading

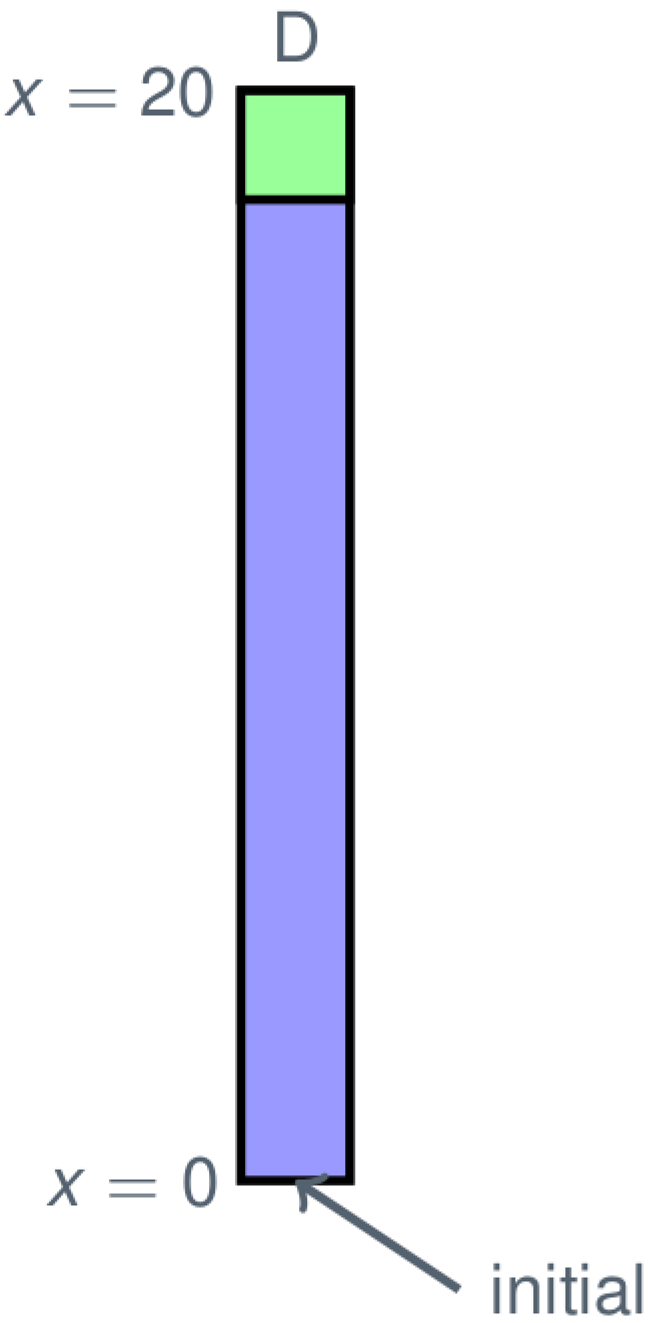


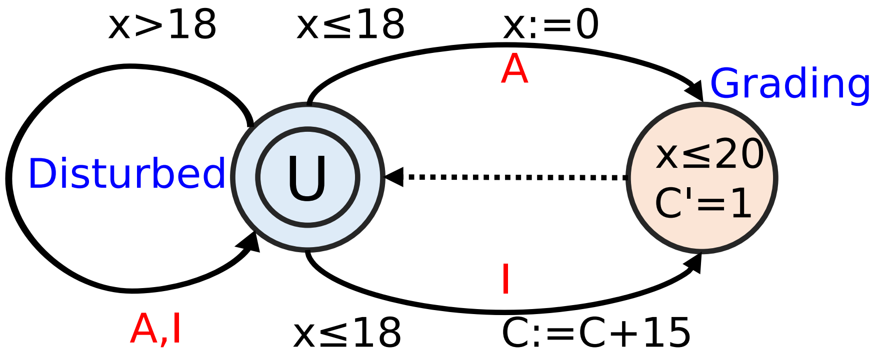
Needs answers

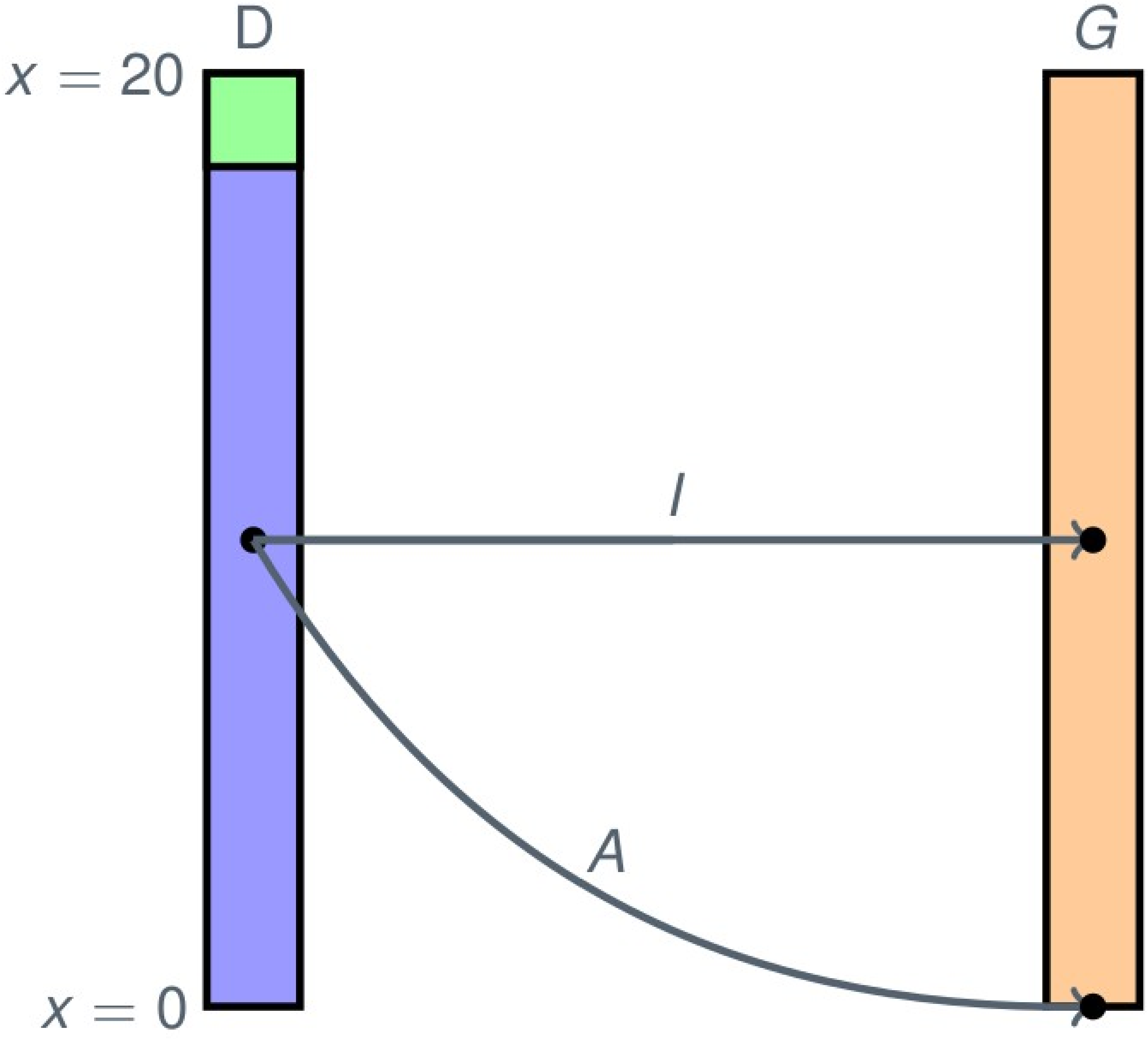
Controllable Professor

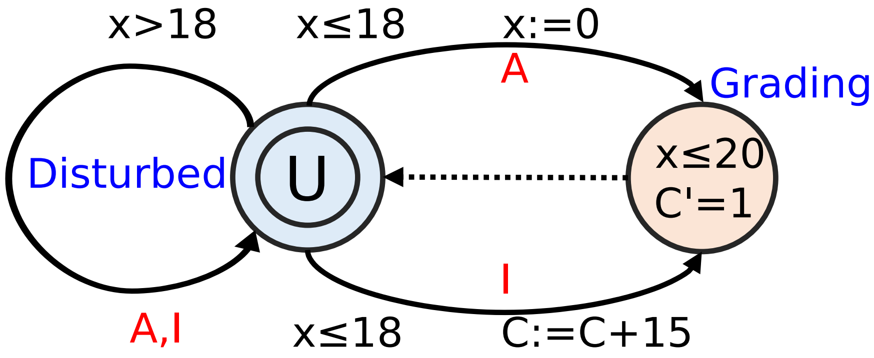
Uncontrollable Student

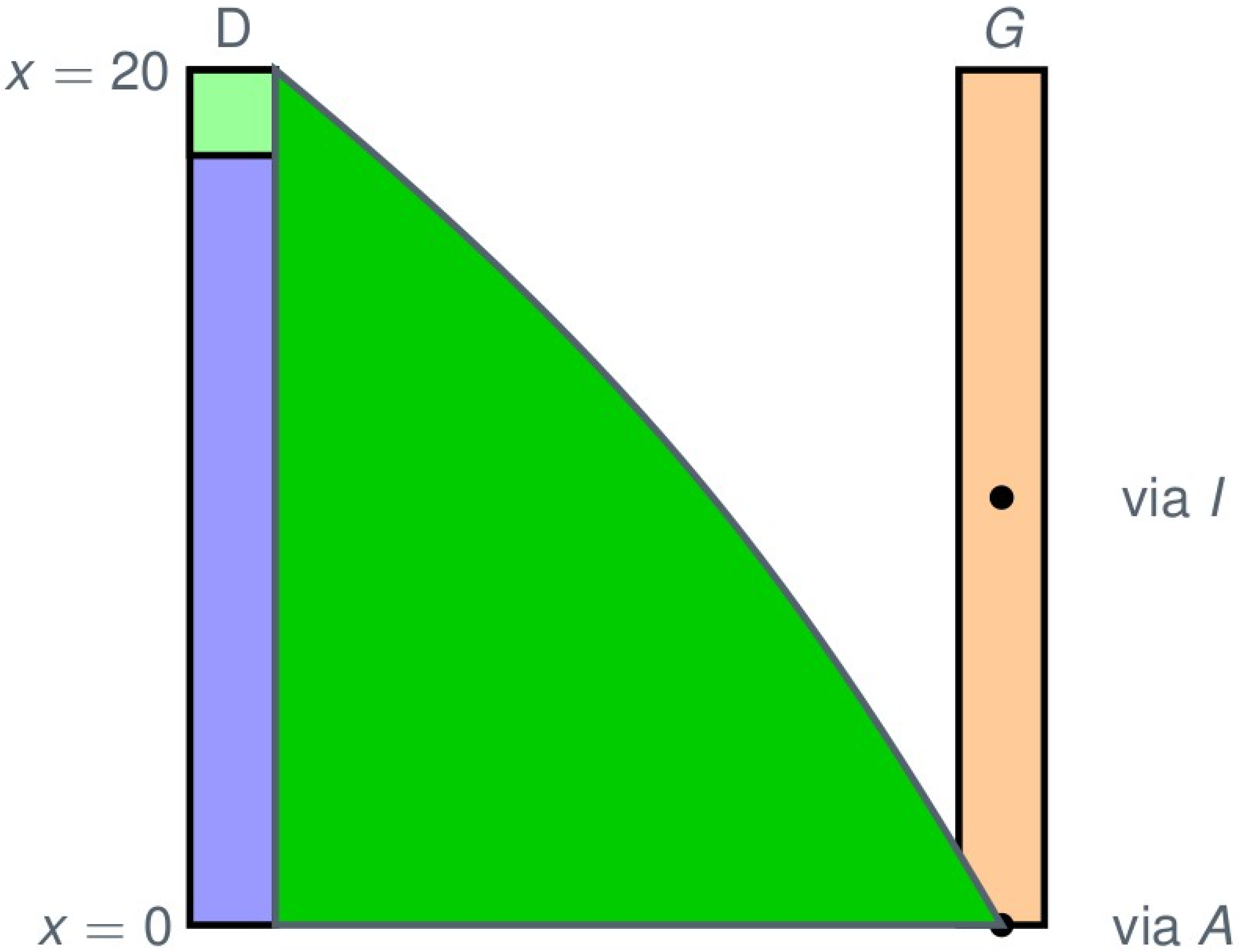
Choices:
1) Answer student, restart grading
2) Ignore the student, spend 15 hours on complaint later

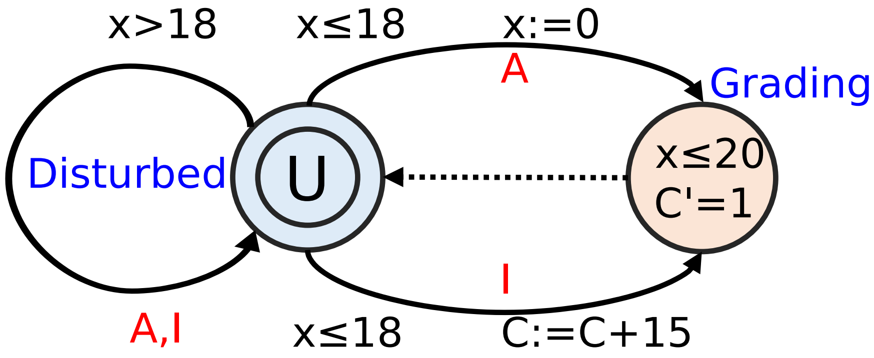

D
G
x = 20
x = 0
initial


x>18
x≤18
x:=0
A
Grading
Disturbed
U
x≤20
C'=1
A,I
I
x≤18
C:=C+15

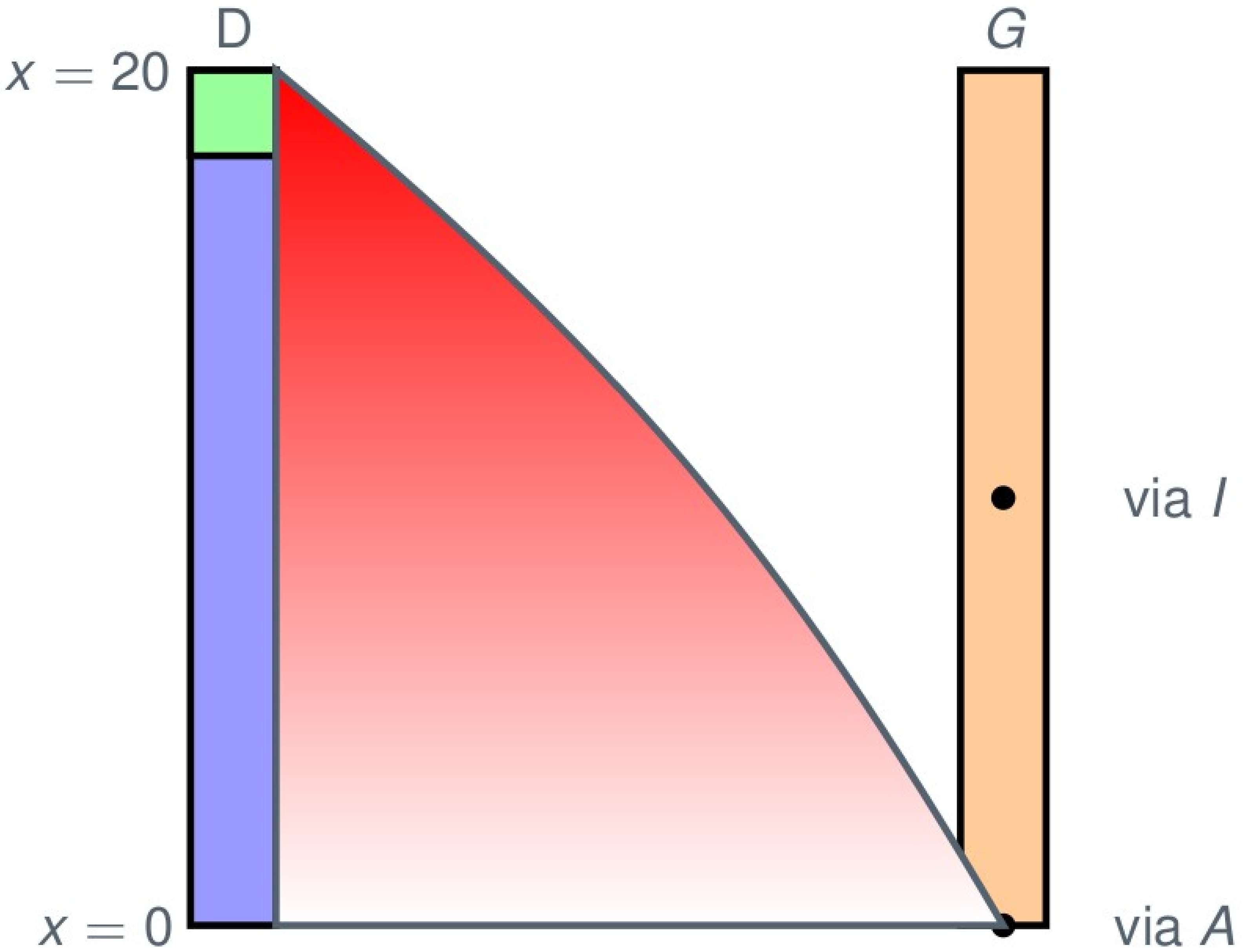

D
G
$x = 20$
$x = 0$
I
A

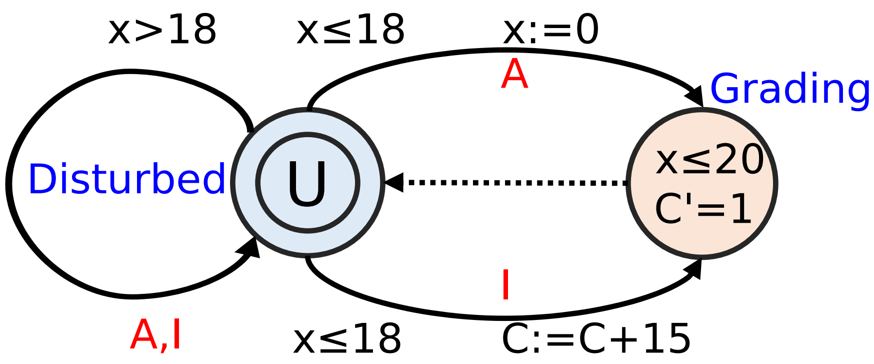

x>18
x≤18
x:=0
A
Grading
Disturbed
U
x≤20
C'=1
I
A,I
x≤18
C:=C+15





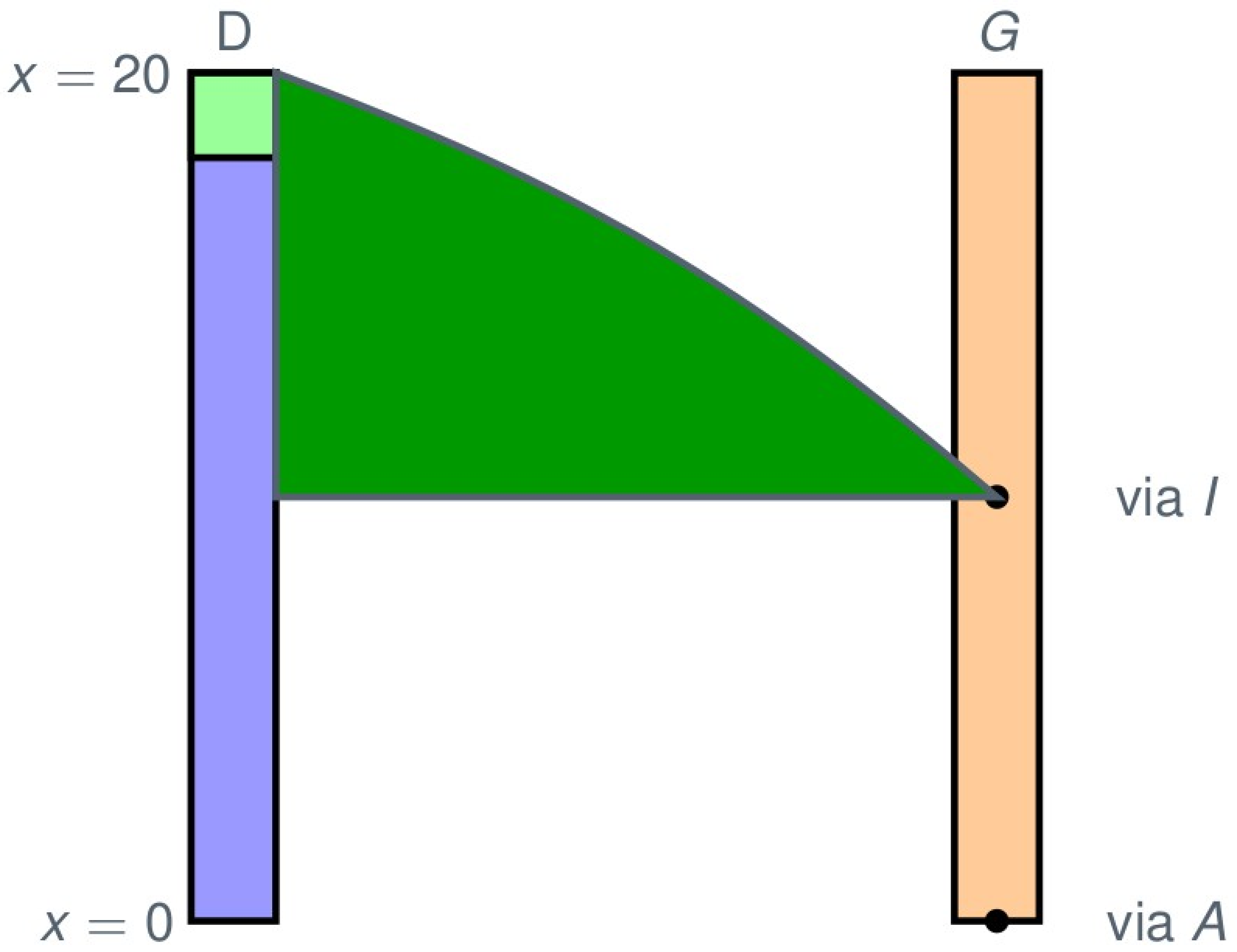
next x = *uniform*$(0, 20)$

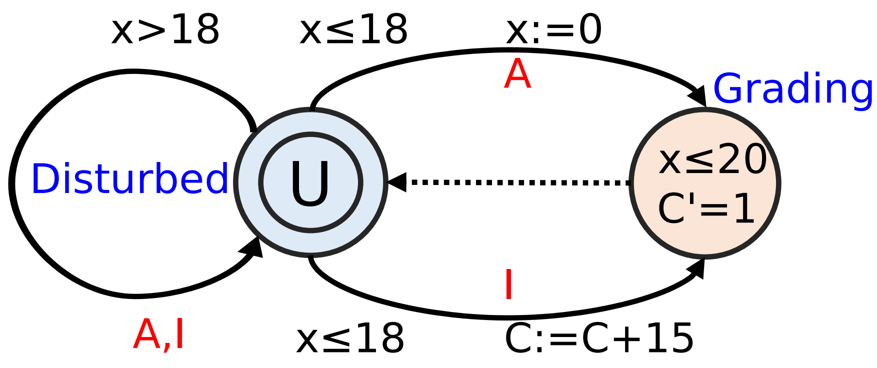




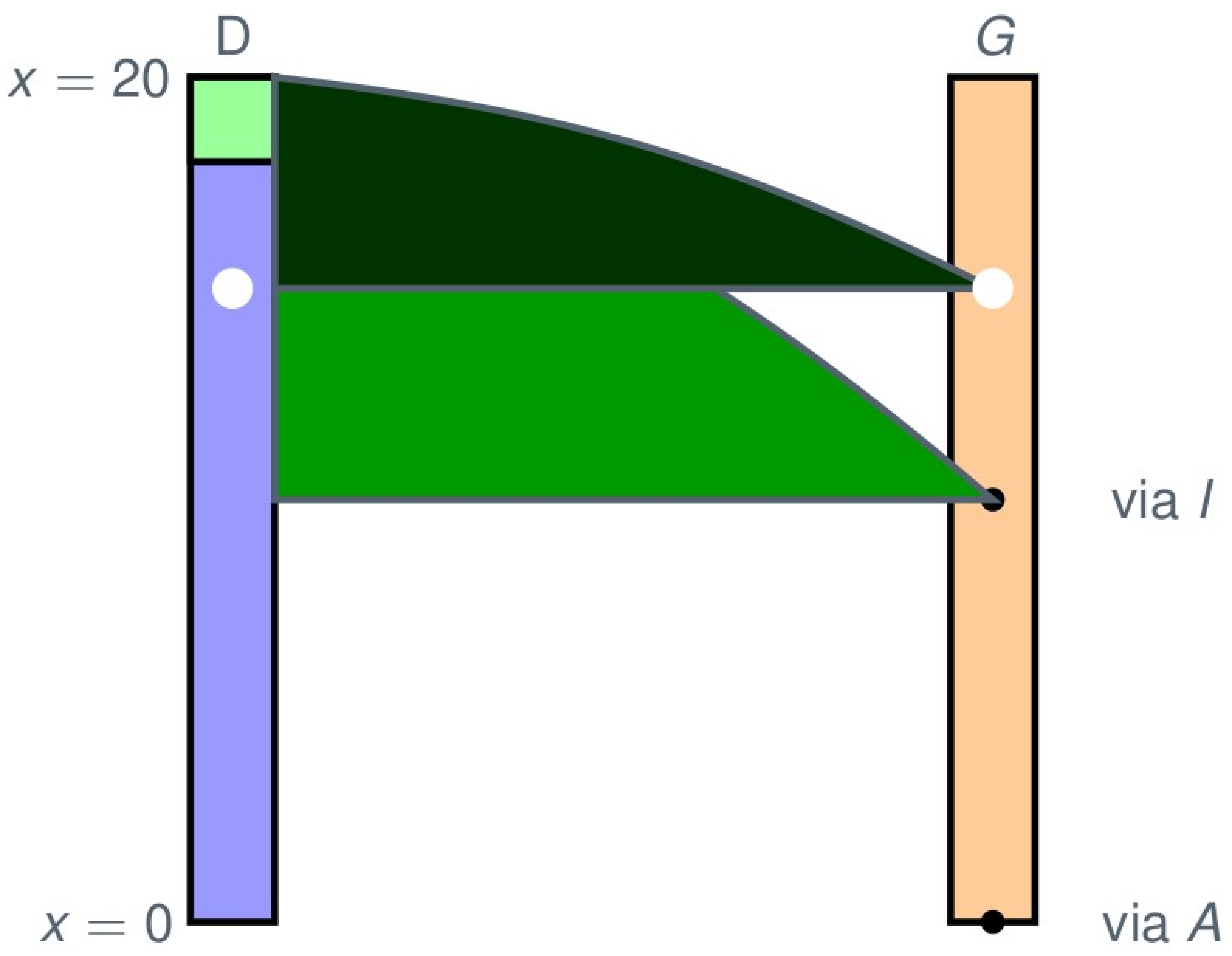
next x = *uniform*(0, 20)

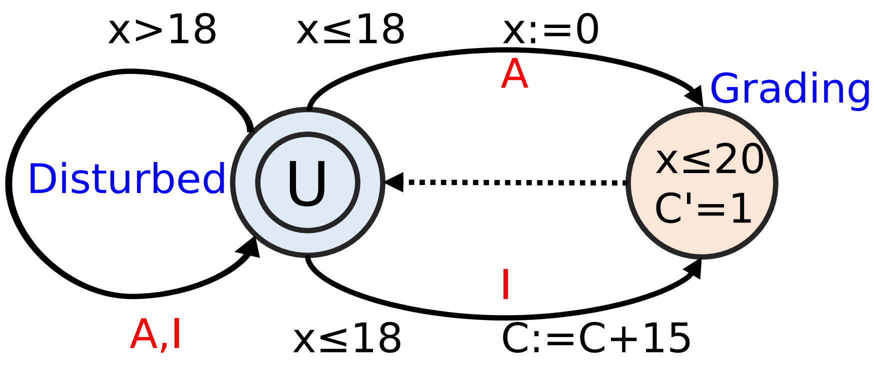
next $C = C +$ *next* x





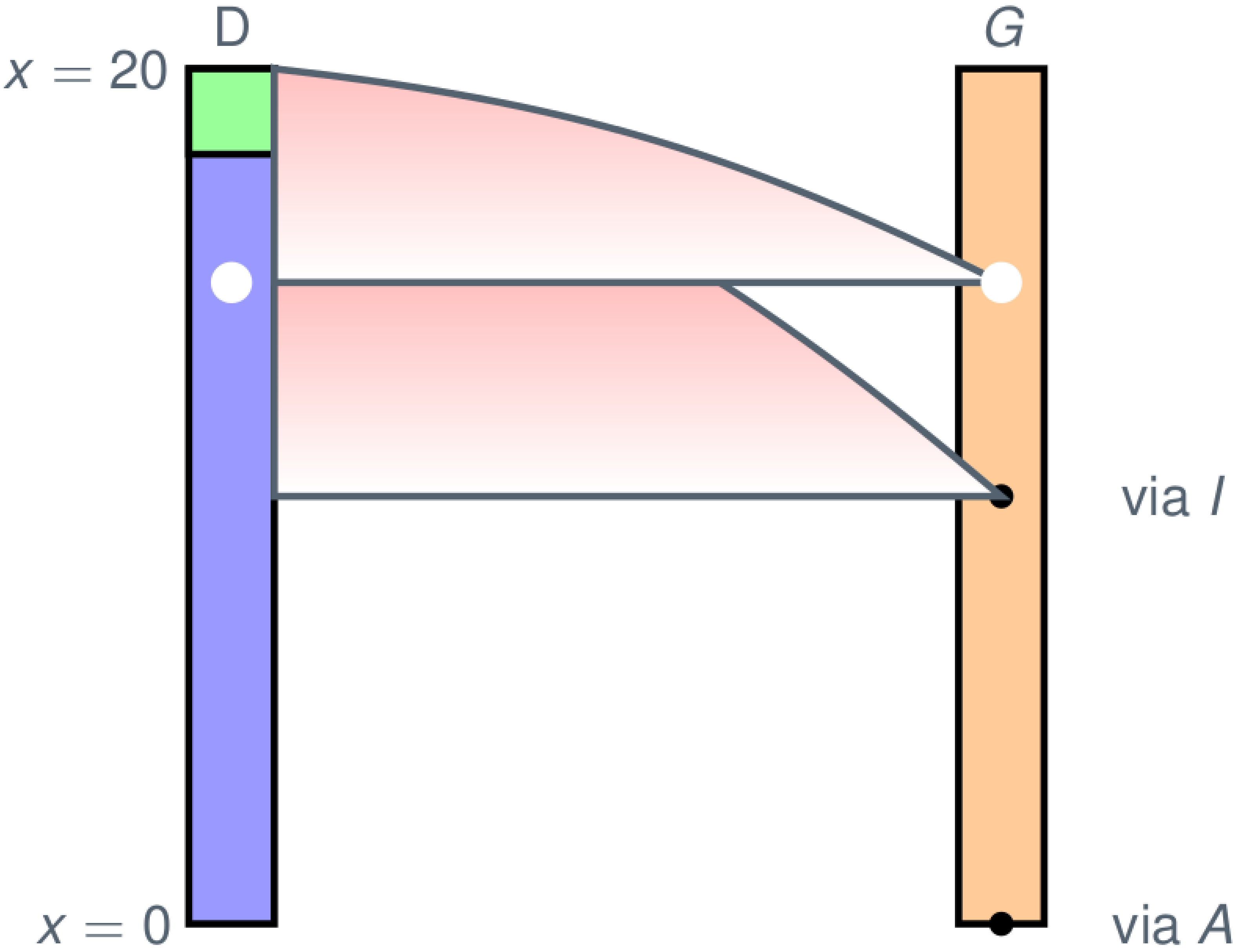
next x = *uniform*$(10, 20)$

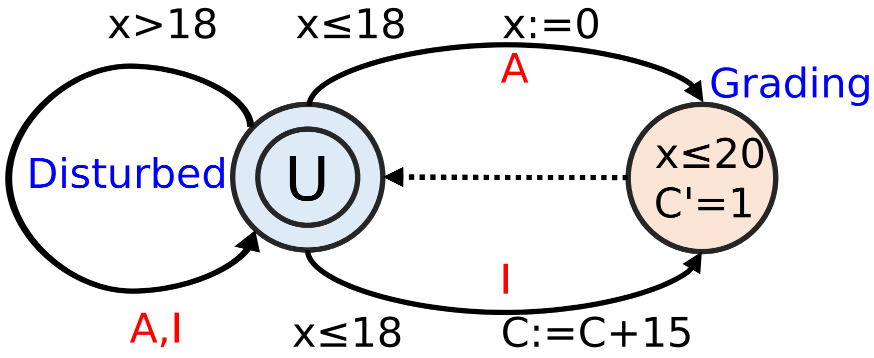




next x = $uniform(10, 20)$

next x = $uniform(15, 20)$





$\text{next } C = C + \text{next } x$

relative to x

Euclidean Markov Decision Process
MDP With Continuous State Space

Undecidable to solve in theory and practice

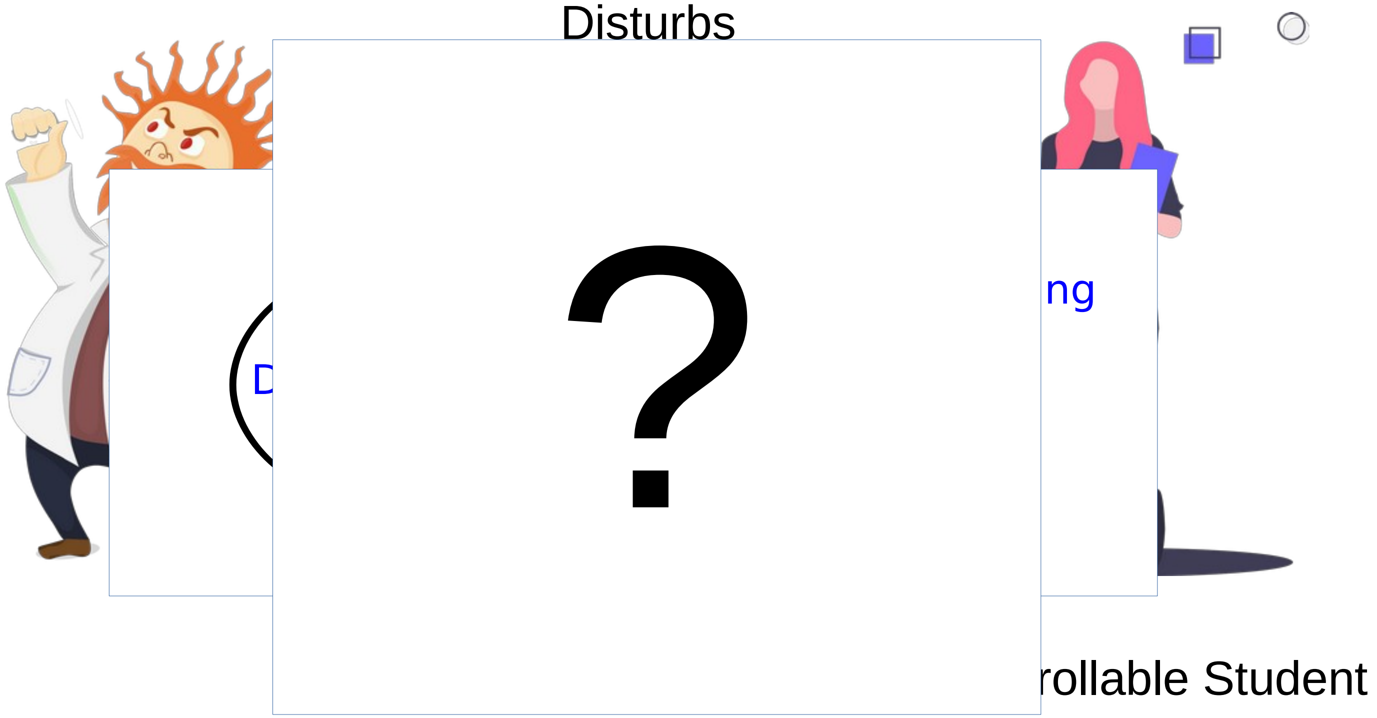
- Infinite (uncountable) number of states
- We cannot map to an regular MDP

Unobservable

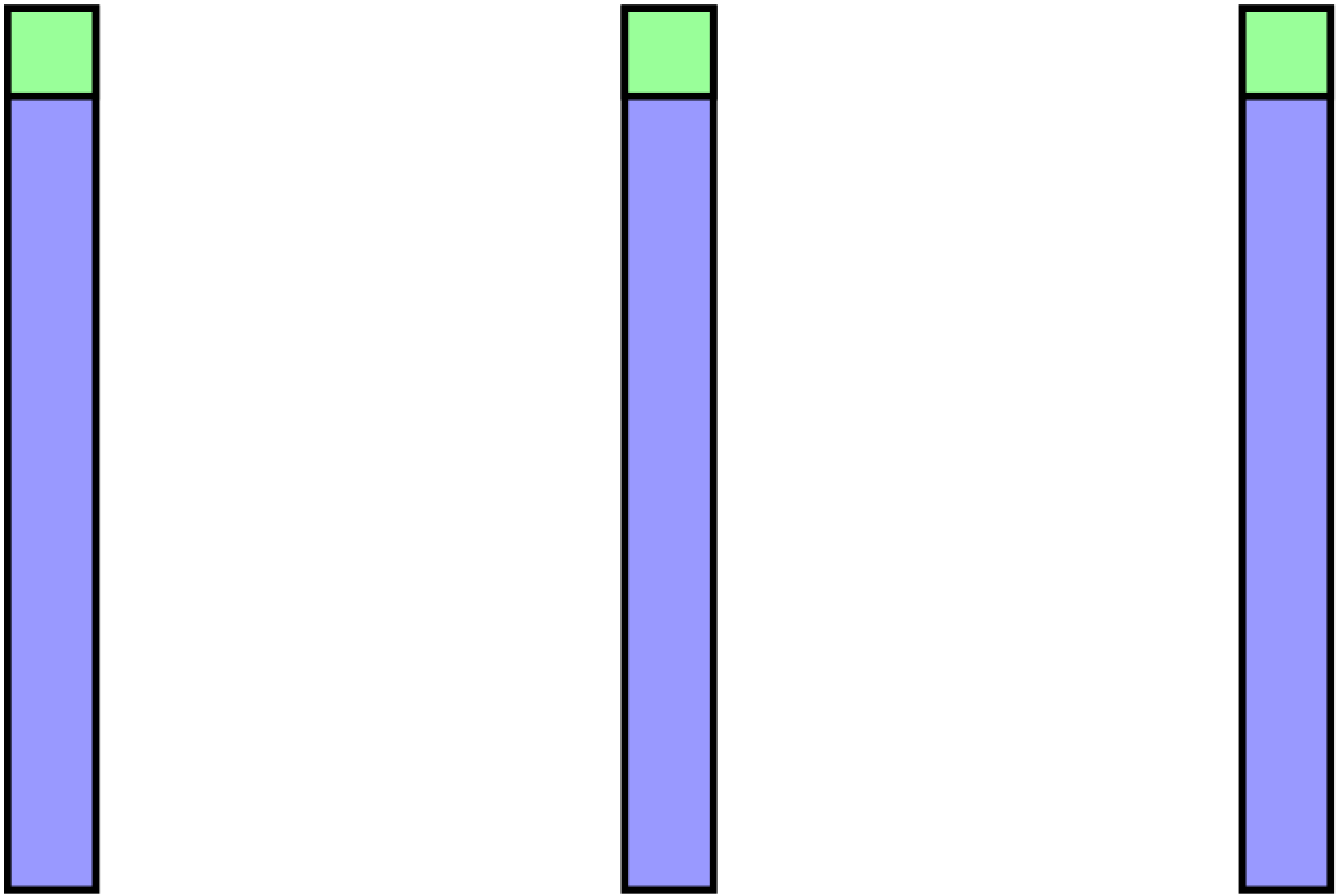
- No direct cost function
- No direct transition function

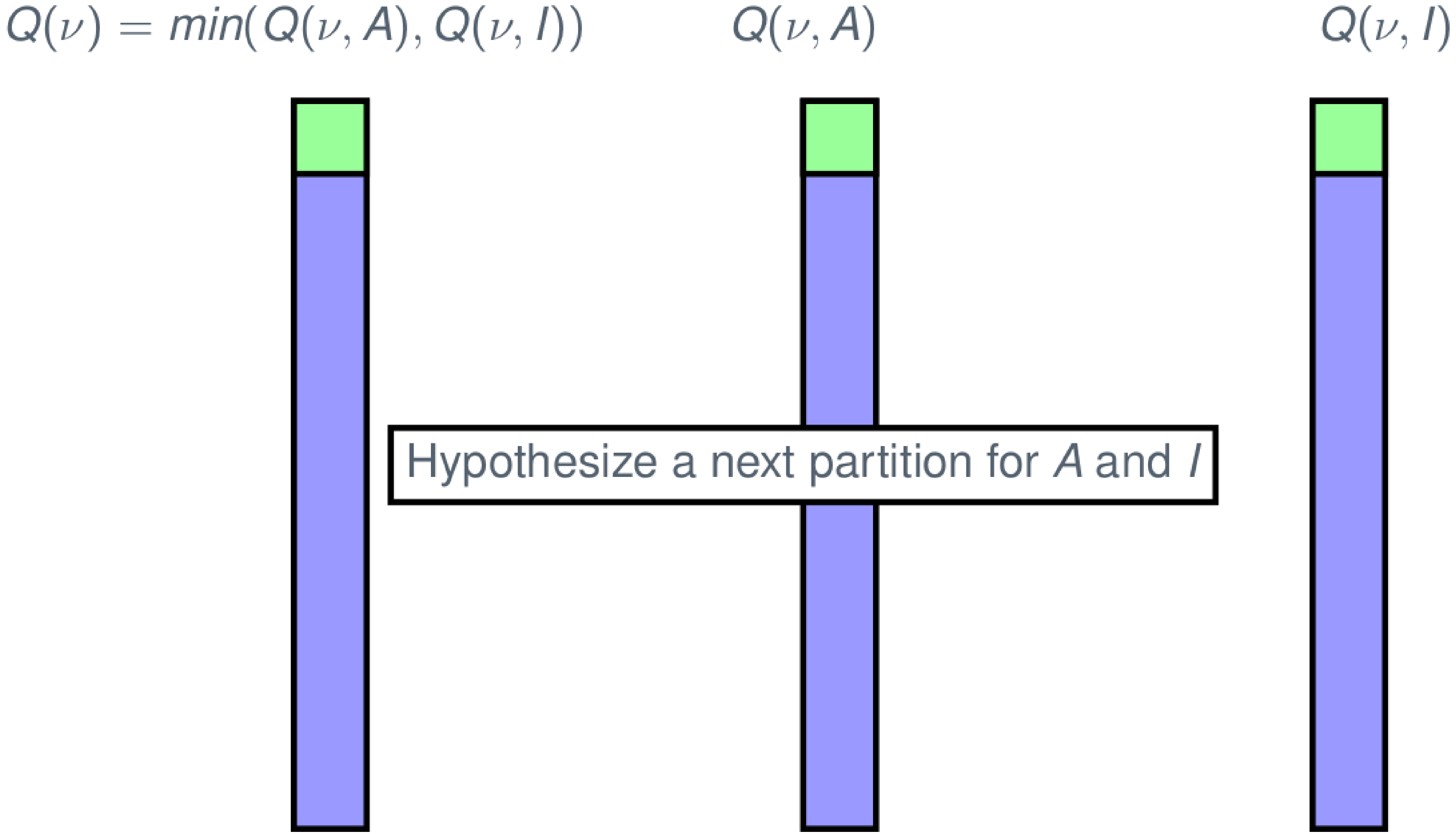
Solution

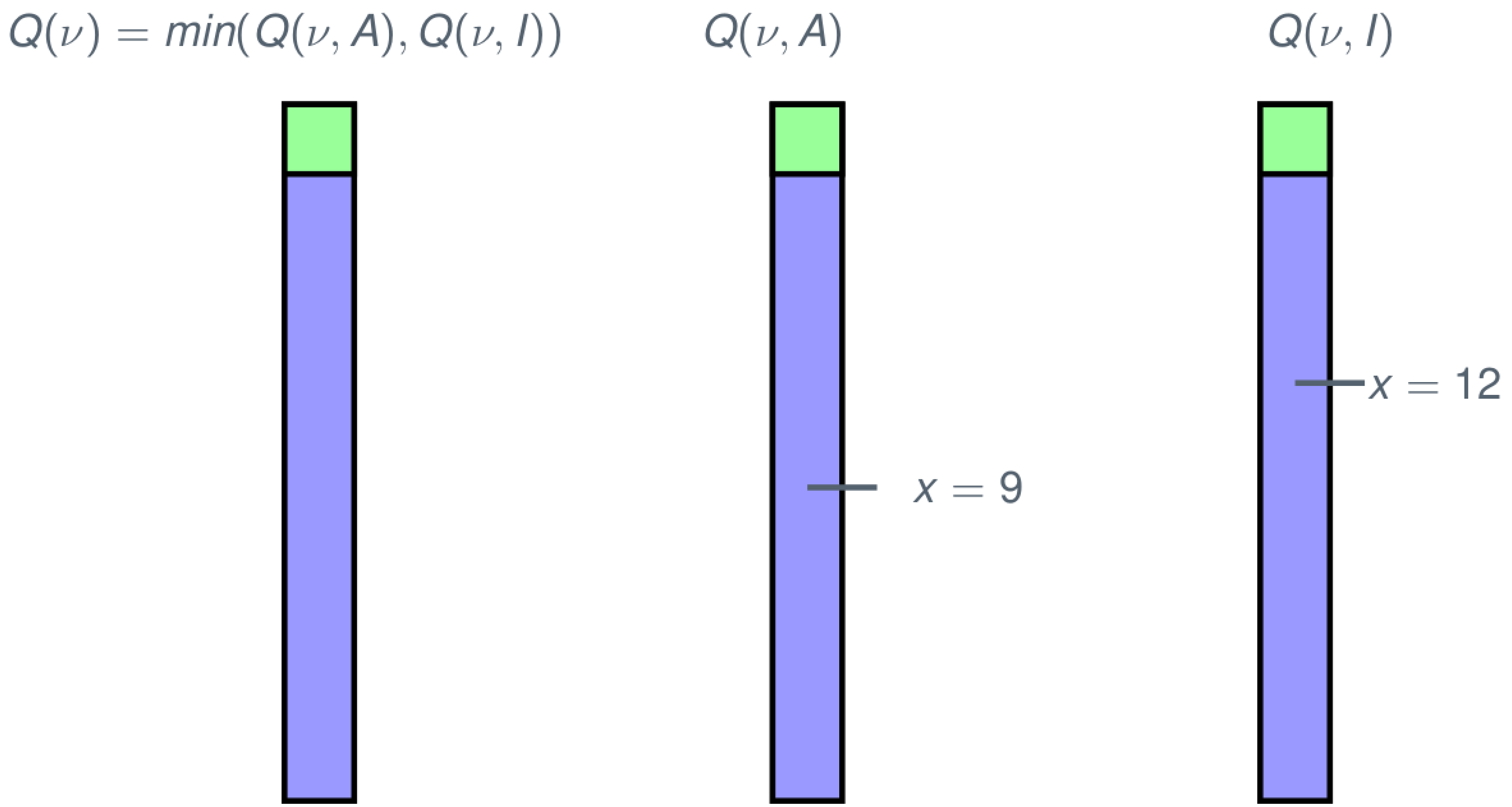
- Discretize!
- What is a good general discretization?

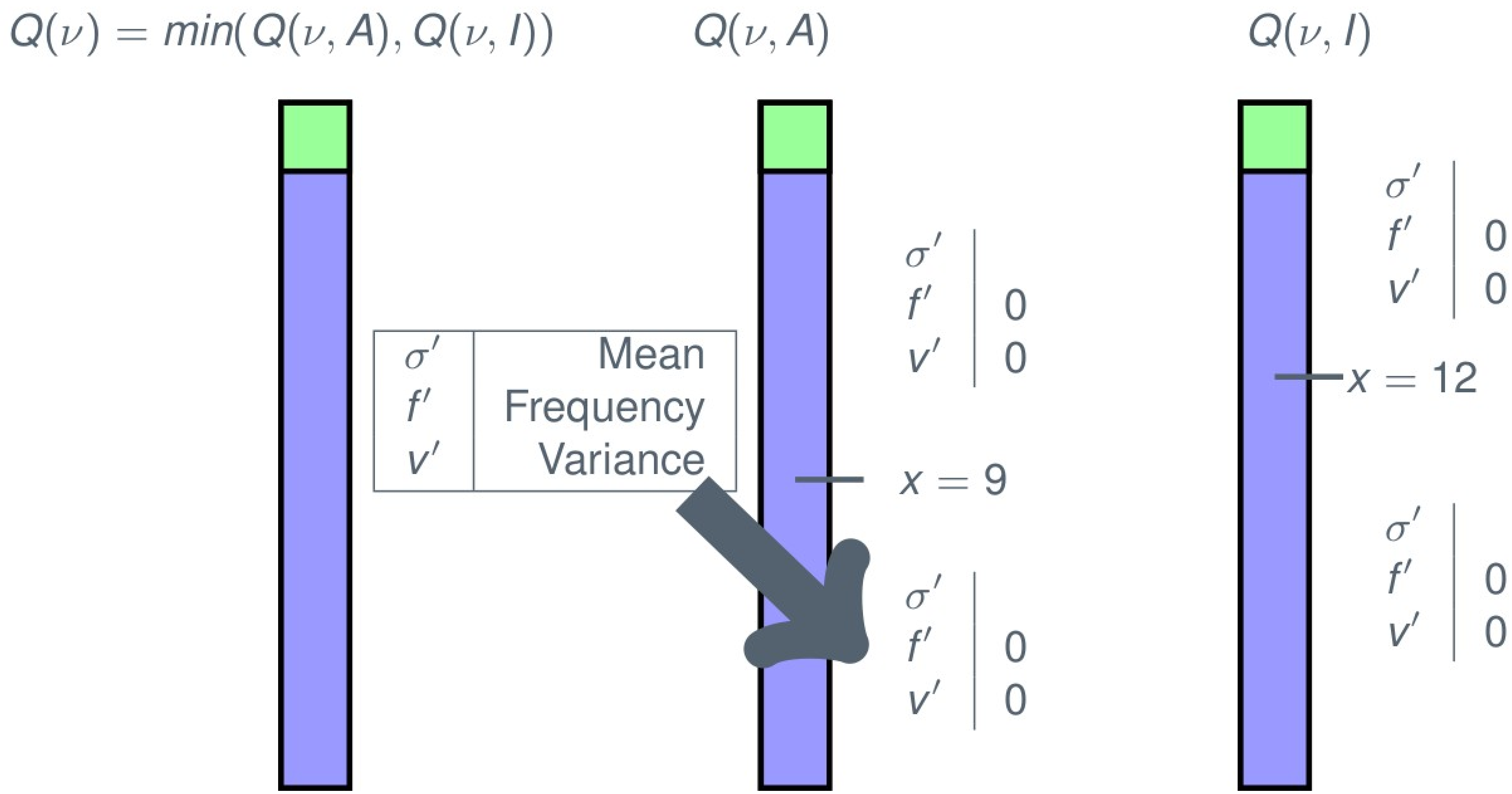


$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$ $\quad$ $Q(\nu, A)$ $\quad$ $Q(\nu, I)$

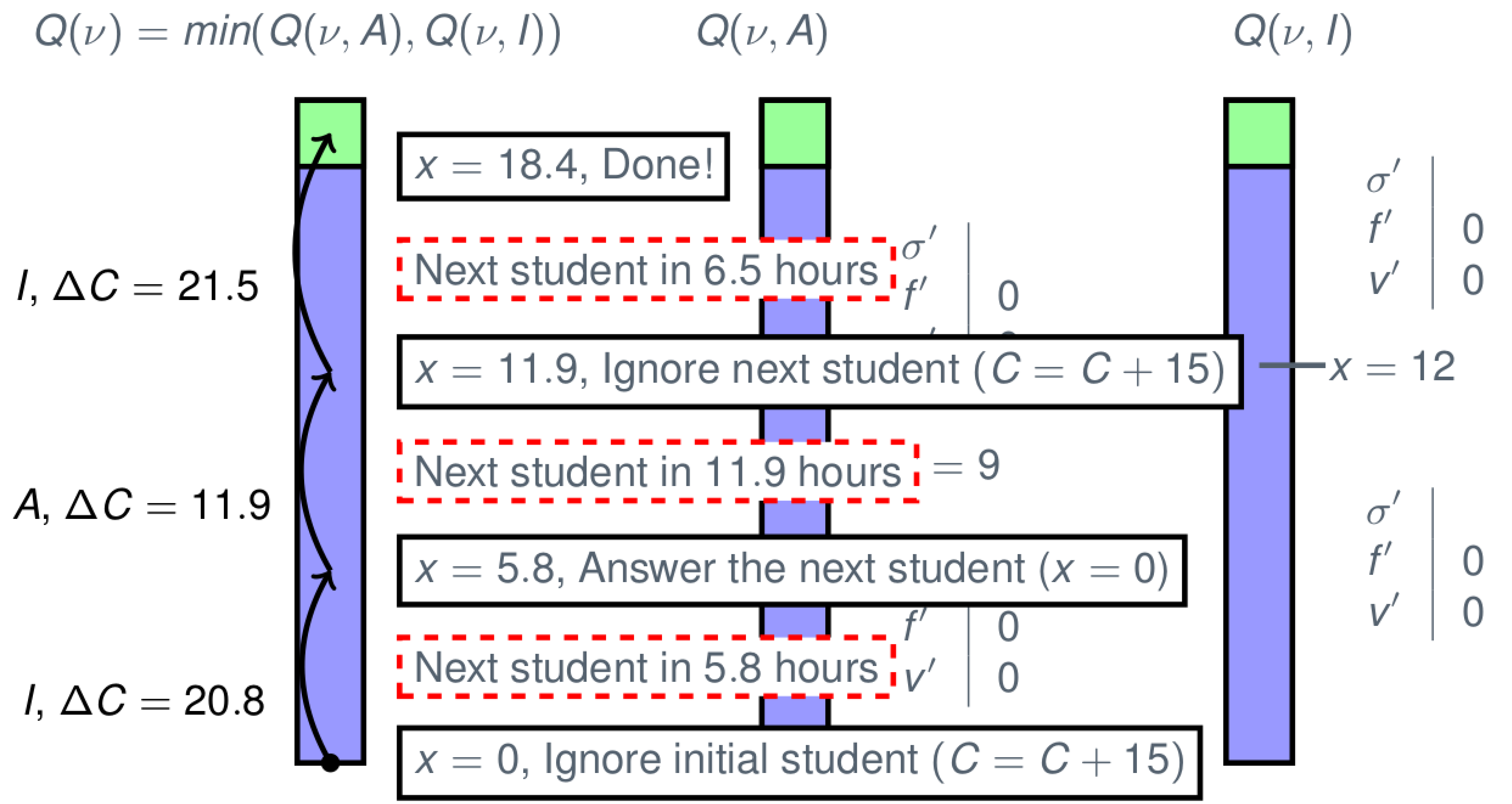



$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
Hypothesize a next partition for A and I

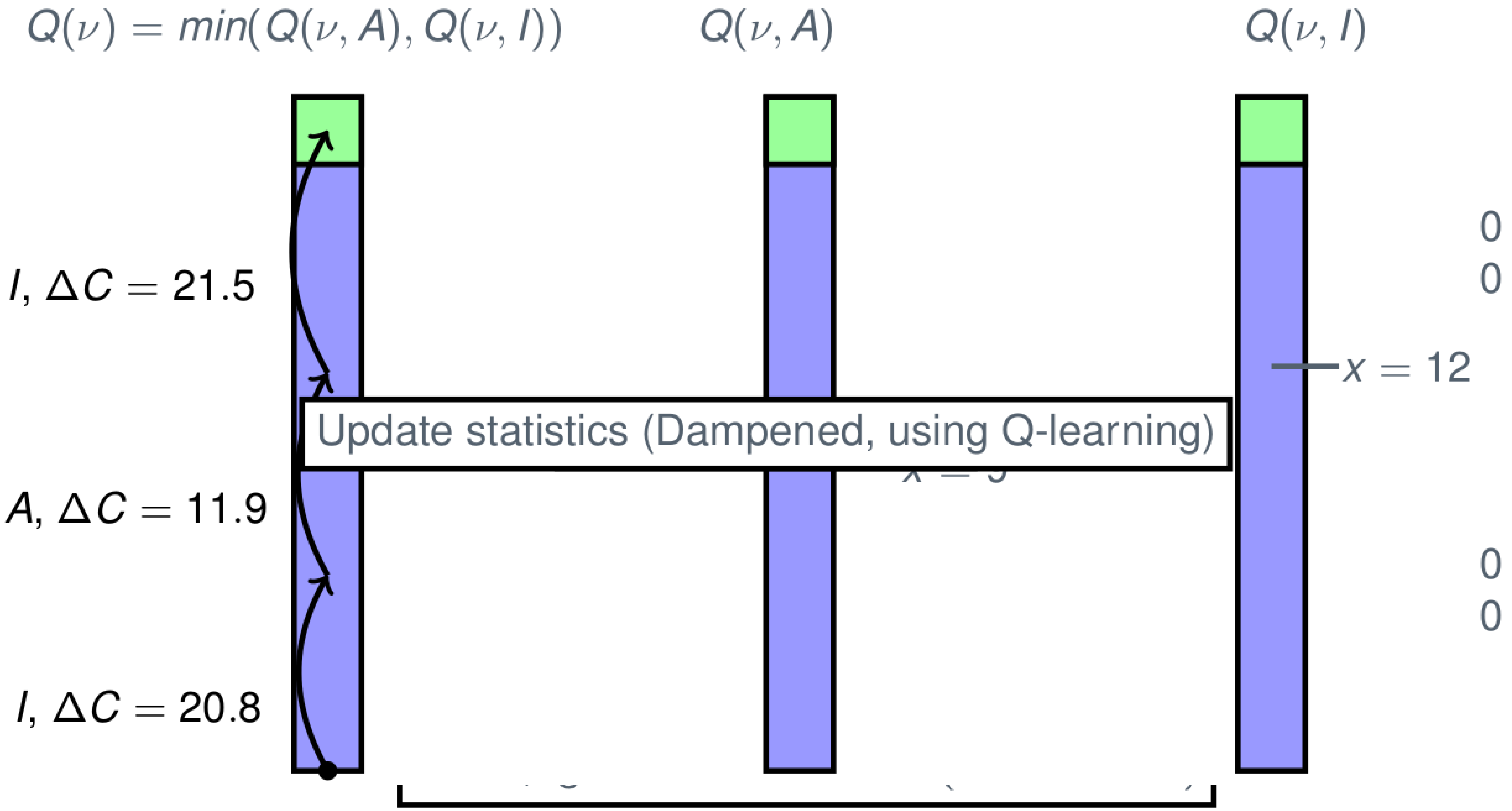

$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$x = 9$
$x = 12$



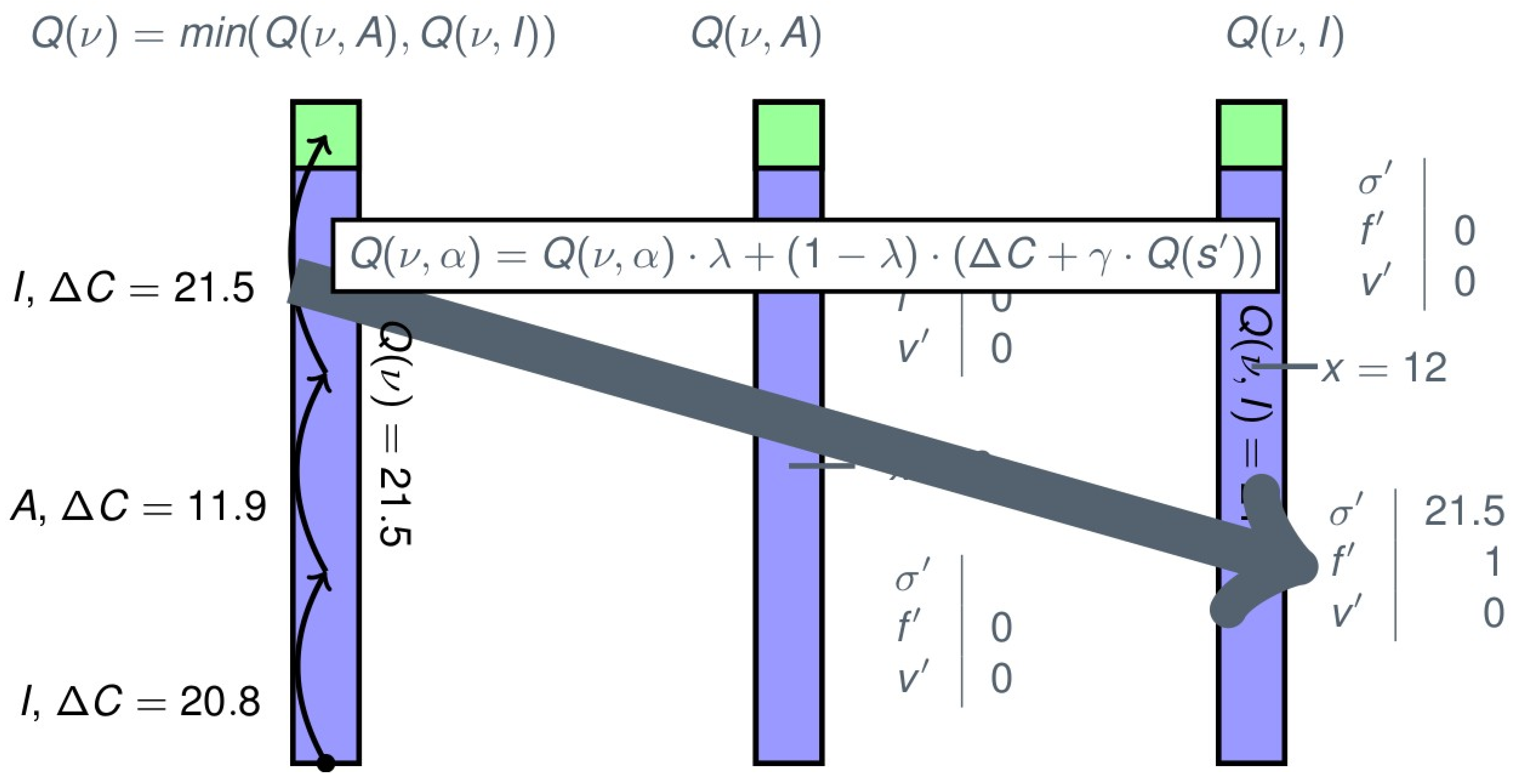
Q(ν) = min(Q(ν, A), Q(ν, I))
Q(ν, A)
Q(ν, I)
σ′ Mean
f′ Frequency
v′ Variance
σ′
f′ 0
v′ 0
x = 9
σ′
f′ 0
v′ 0
σ′
f′ 0
v′ 0
x = 12
σ′
f′ 0
v′ 0



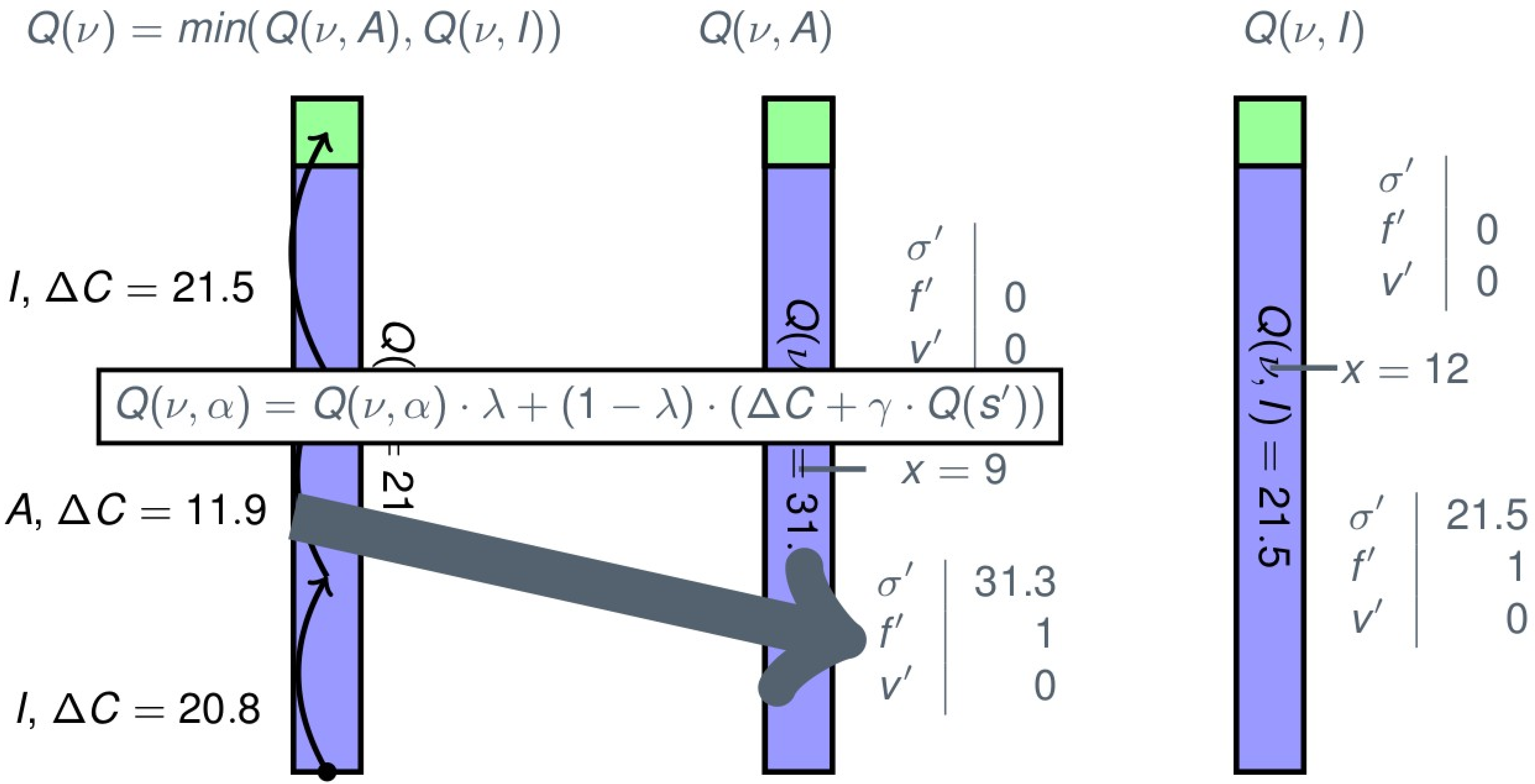
$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$x = 18.4$, Done!
Next student in 6.5 hours
$x = 11.9$, Ignore next student ($C = C + 15$)
$x = 12$
Next student in 11.9 hours
$= 9$
$x = 5.8$, Answer the next student ($x = 0$)
Next student in 5.8 hours
$x = 0$, Ignore initial student ($C = C + 15$)
$I, \Delta C = 21.5$
$A, \Delta C = 11.9$
$I, \Delta C = 20.8$
σ'
f' 0
v' 0



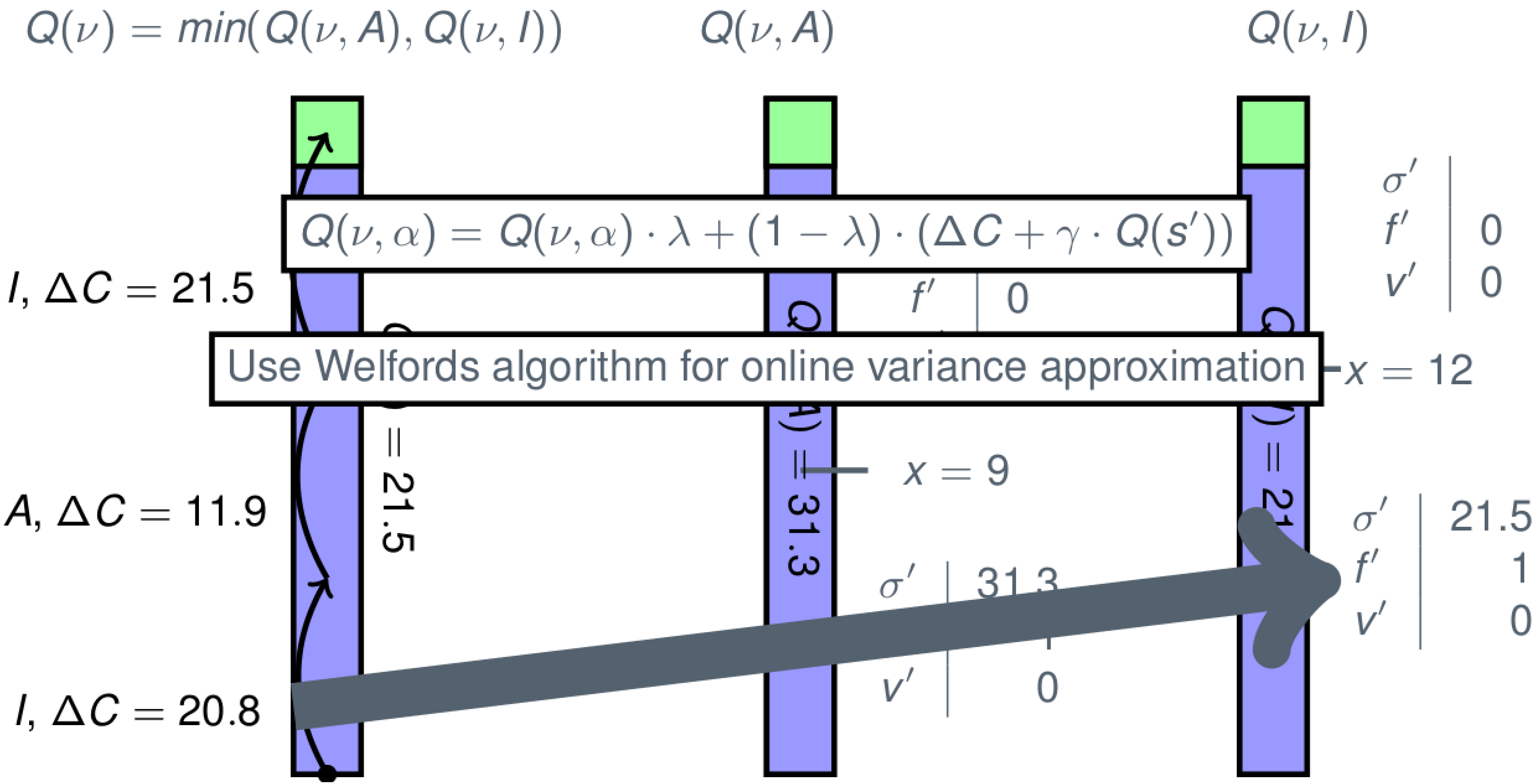
$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$I, \Delta C = 21.5$
$A, \Delta C = 11.9$
$I, \Delta C = 20.8$
$x = 12$
0
0
0
0
Update statistics (Dampened, using Q-learning)



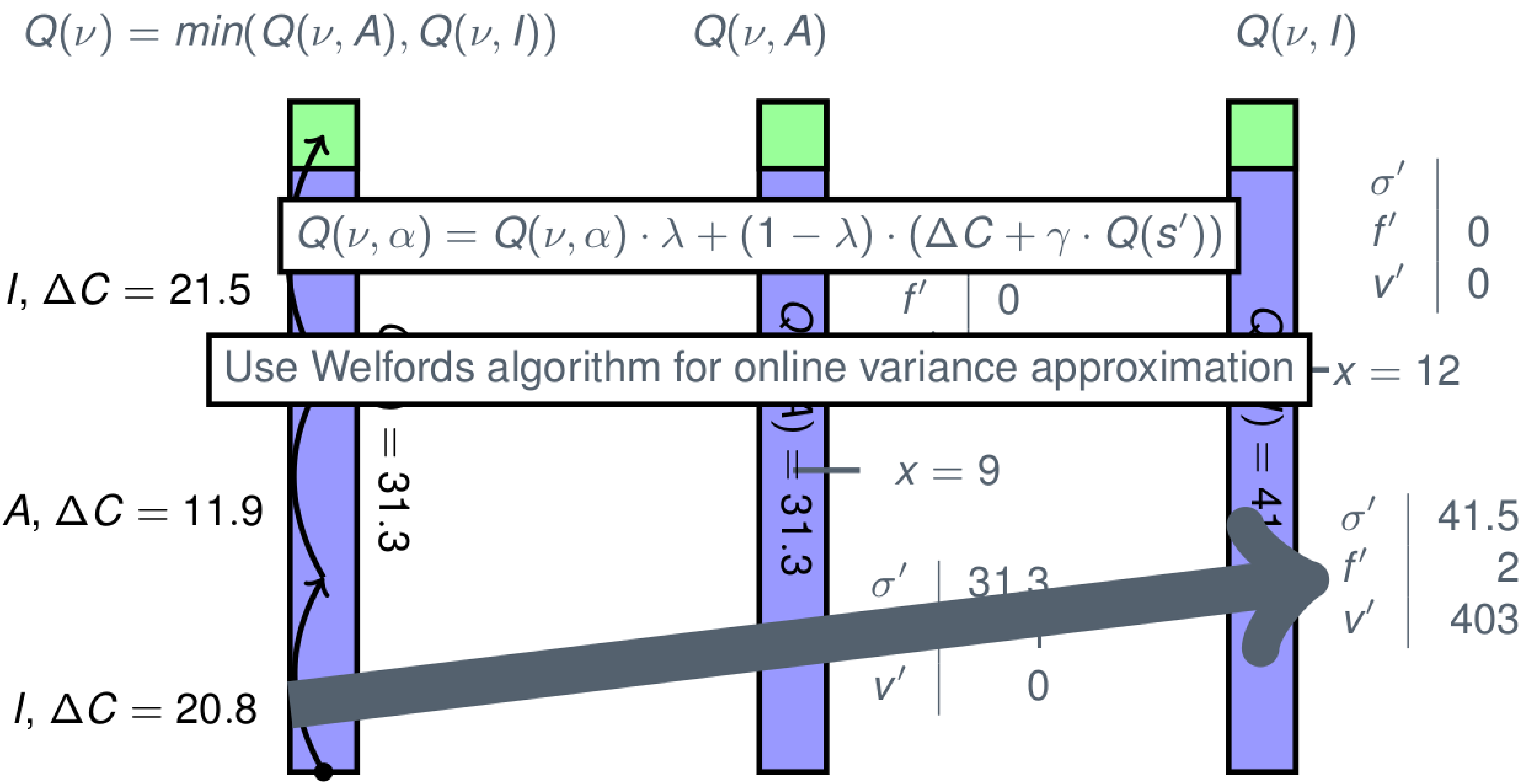
$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$Q(\nu, \alpha) = Q(\nu, \alpha) \cdot \lambda + (1 - \lambda) \cdot (\Delta C + \gamma \cdot Q(s'))$
$I, \Delta C = 21.5$
$A, \Delta C = 11.9$
$I, \Delta C = 20.8$
$Q(\nu) = 21.5$
$x = 12$
σ'
f' 0
v' 0
v' 0
σ' 21.5
f' 1
v' 0
σ'
f' 0
v' 0



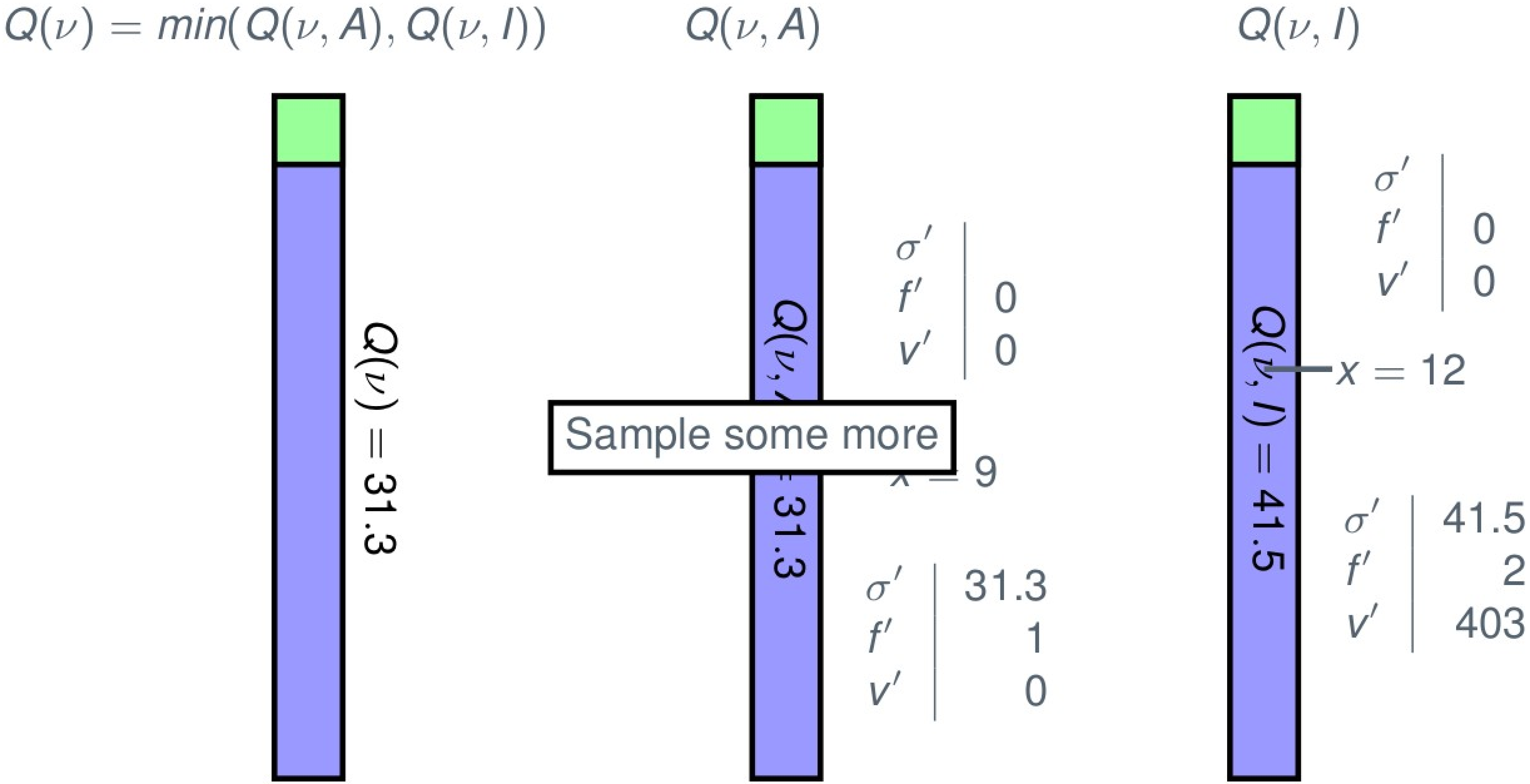
$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$I, \Delta C = 21.5$
$A, \Delta C = 11.9$
$I, \Delta C = 20.8$
$Q(\nu, \alpha) = Q(\nu, \alpha) \cdot \lambda + (1 - \lambda) \cdot (\Delta C + \gamma \cdot Q(s'))$
σ'
f' 0
v' 0
$x = 9$
σ' 31.3
f' 1
v' 0
σ'
f' 0
v' 0
$x = 12$
$Q(\nu, I) = 21.5$
σ' 21.5
f' 1
v' 0



$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$Q(\nu, \alpha) = Q(\nu, \alpha) \cdot \lambda + (1 - \lambda) \cdot (\Delta C + \gamma \cdot Q(s'))$
Use Welfords algorithm for online variance approximation
$I, \Delta C = 21.5$
$A, \Delta C = 11.9$
$I, \Delta C = 20.8$
$= 21.5$
$= 31.3$
$x = 9$
$x = 12$
σ' | 31.3
f' | 0
v' | 0
σ'
f' | 0
v' | 0
σ' | 21.5
f' | 1
v' | 0

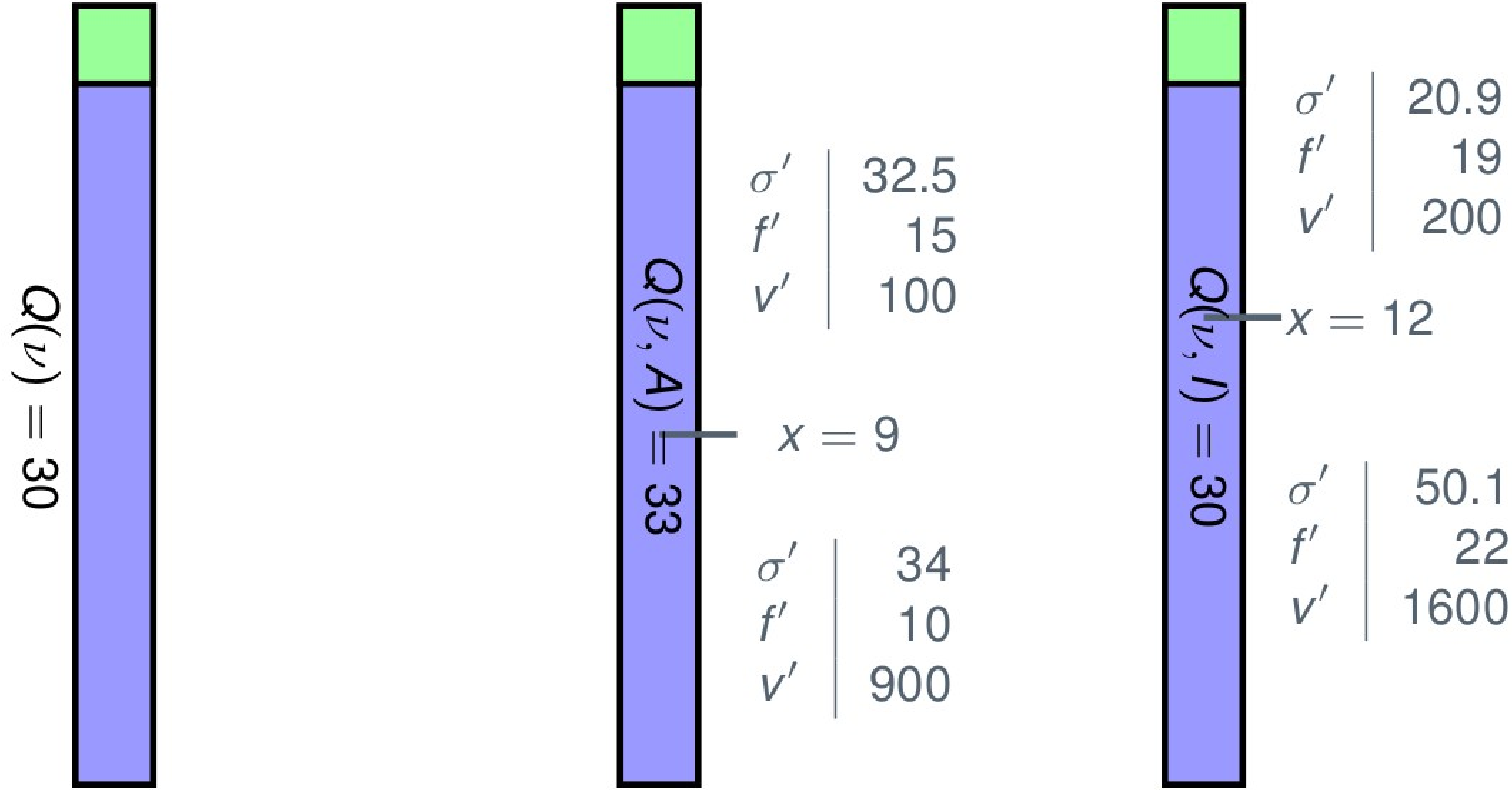


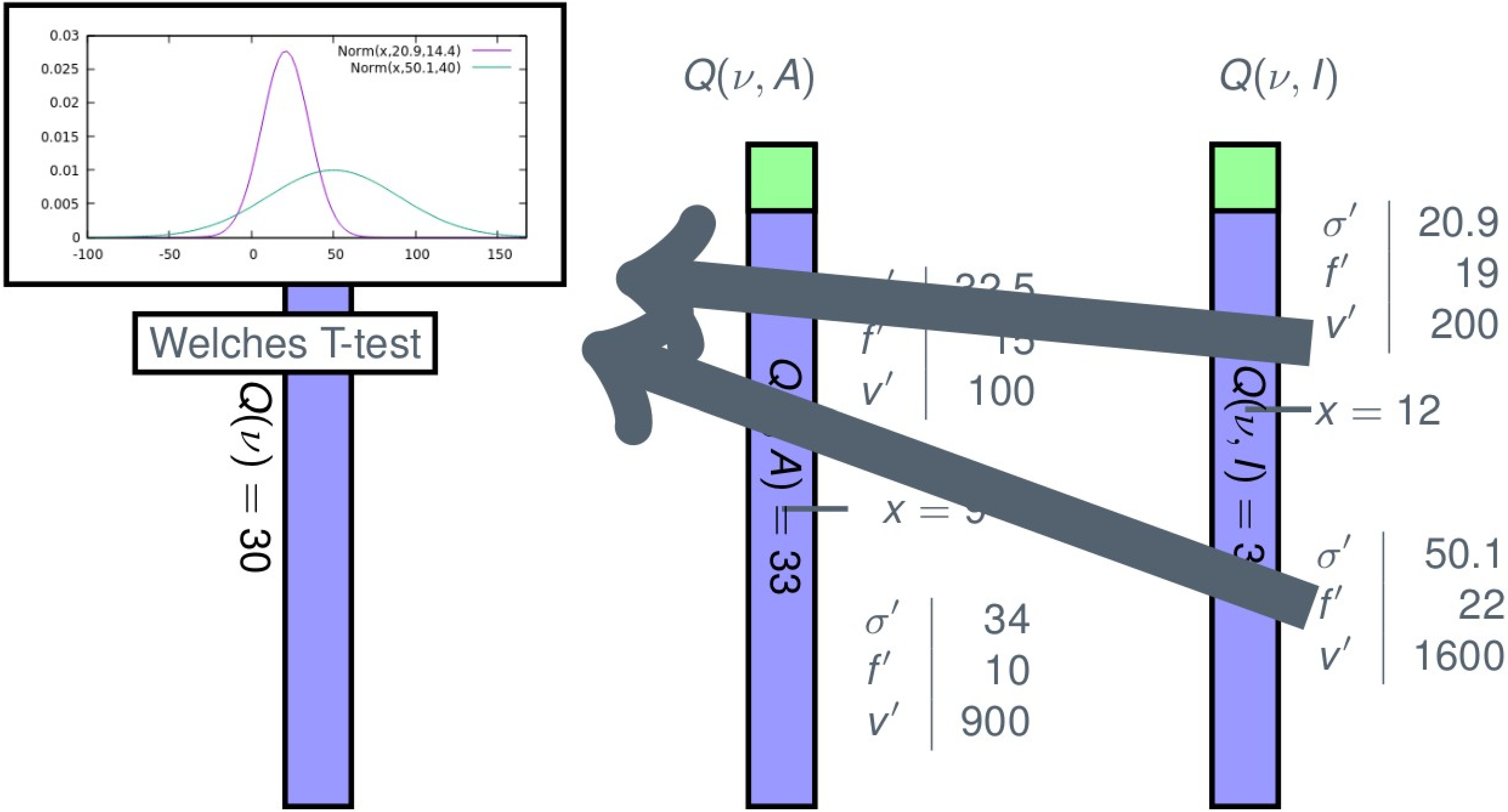
$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$Q(\nu, \alpha) = Q(\nu, \alpha) \cdot \lambda + (1 - \lambda) \cdot (\Delta C + \gamma \cdot Q(s'))$
σ'
f' 0
v' 0
$I, \Delta C = 21.5$
f' 0
Use Welfords algorithm for online variance approximation
$x = 12$
$= 31.3$
$= 31.3$
$x = 9$
$A, \Delta C = 11.9$
σ' 41.5
f' 2
v' 403
σ' 31.3
v' 0
$I, \Delta C = 20.8$



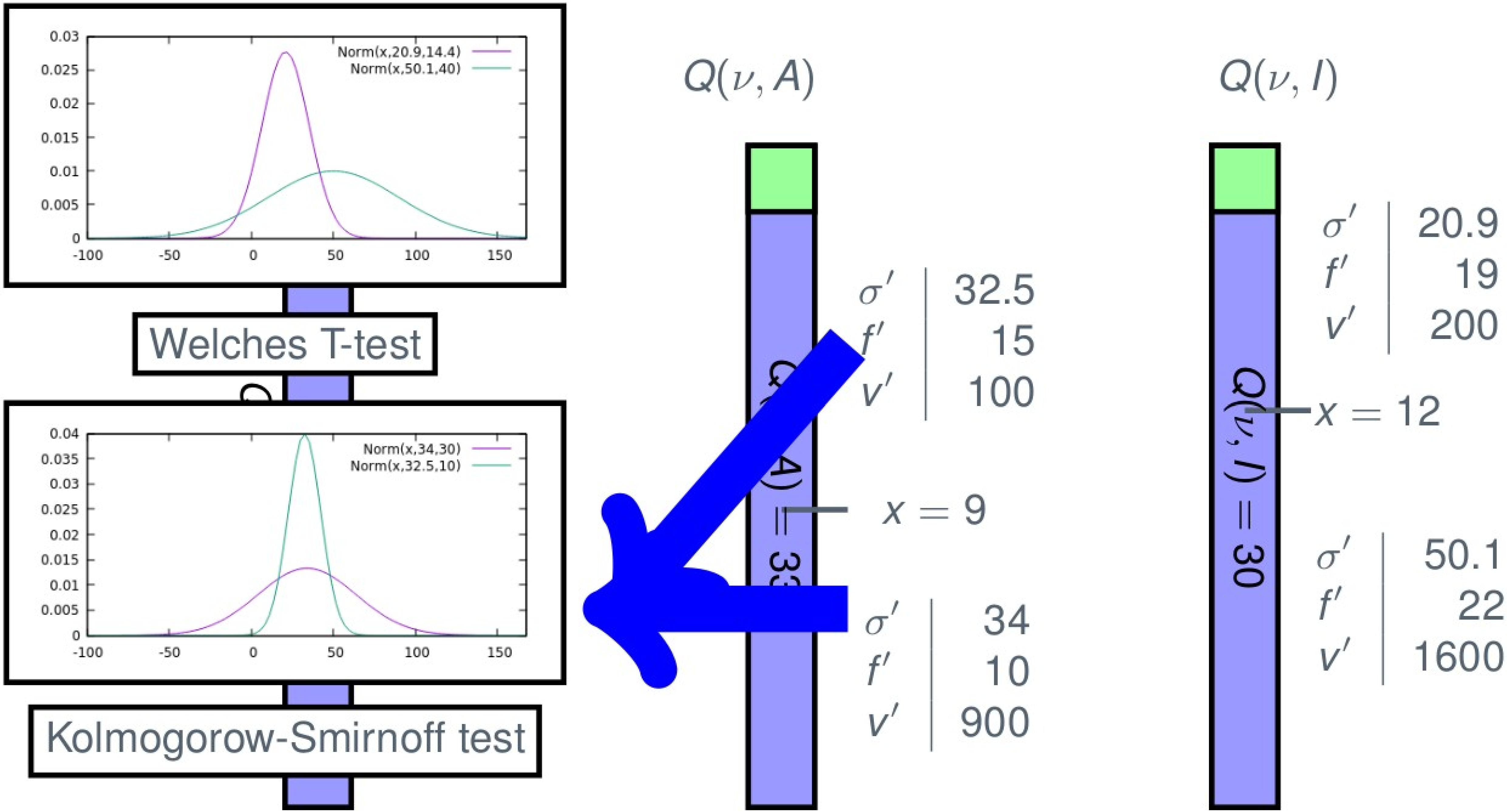
$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$Q(\nu) = 31.3$
$Q(\nu, A) = 31.3$
σ'
f' 0
v' 0
$x = 9$
σ' 31.3
f' 1
v' 0
Sample some more
σ'
f' 0
v' 0
$x = 12$
$Q(\nu, I) = 41.5$
σ' 41.5
f' 2
v' 403

$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$ $\quad$ $Q(\nu, A)$ $\quad$ $Q(\nu, I)$

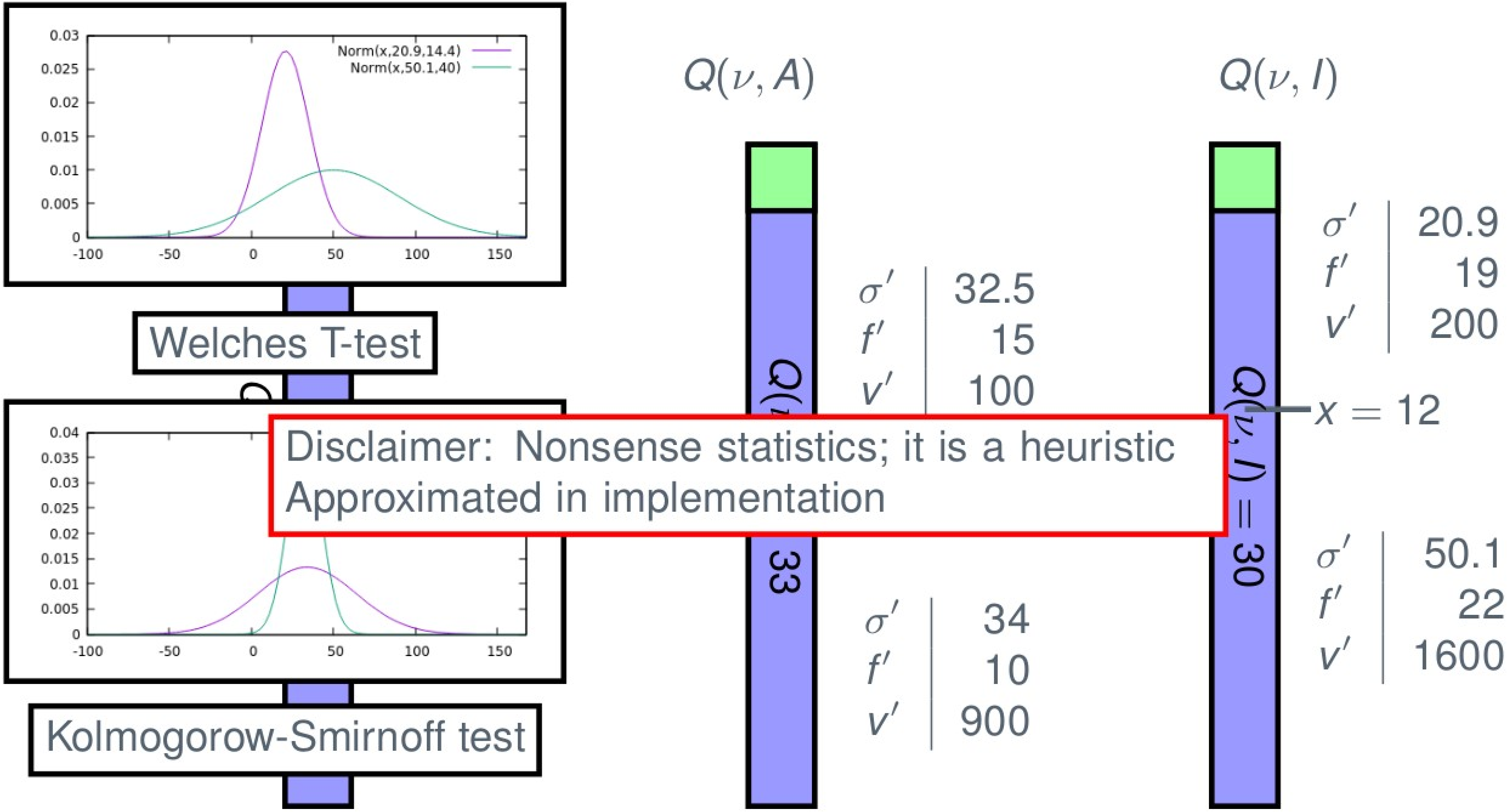


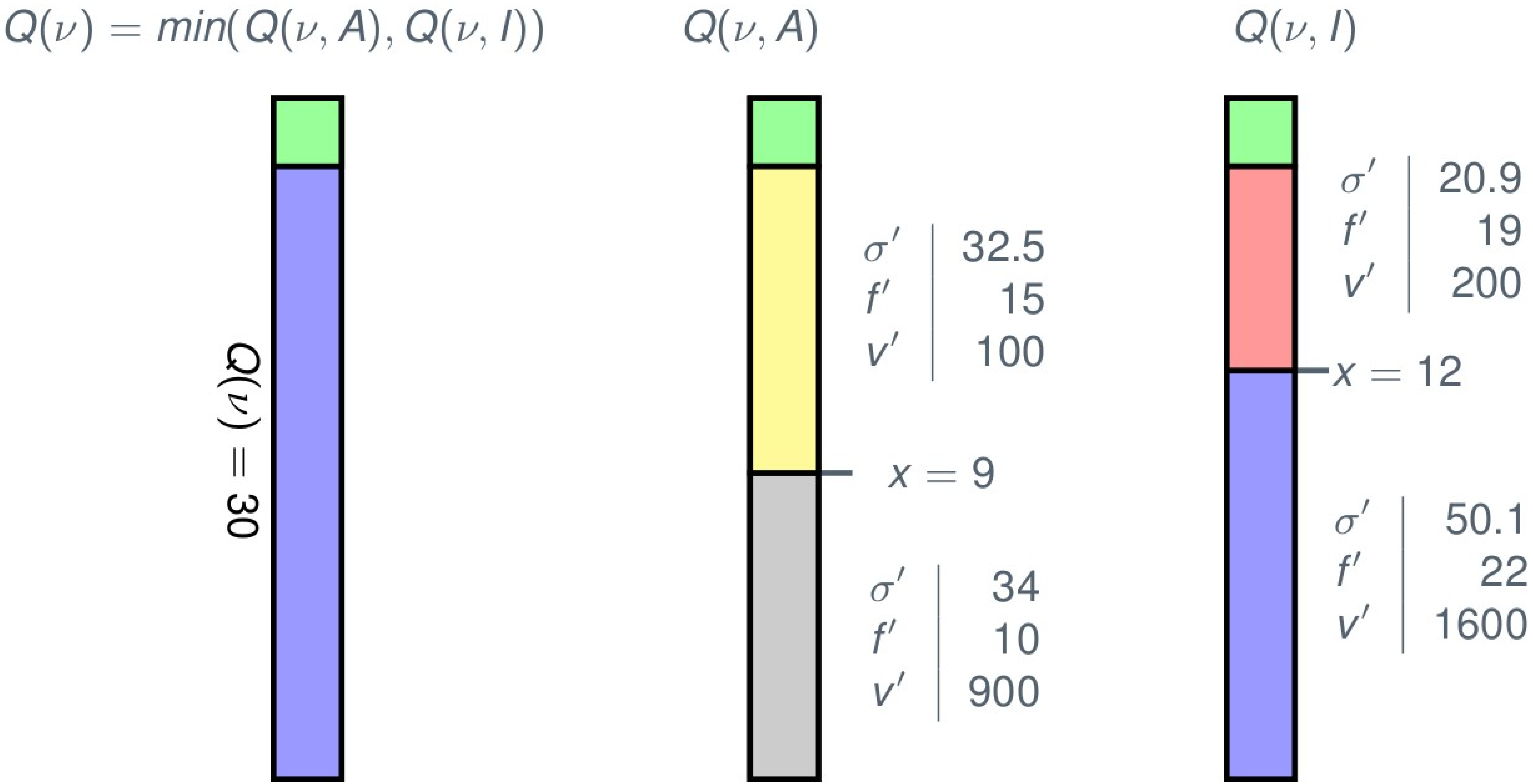


Norm(x,20.9,14.4)
Norm(x,50.1,40)
Welches T-test
$Q(\nu) = 30$
$Q(\nu, A)$
$Q(\nu, I)$
v' 100
σ' 34
f' 10
v' 900
σ' 20.9
f' 19
v' 200
$x = 12$
σ' 50.1
f' 22
v' 1600

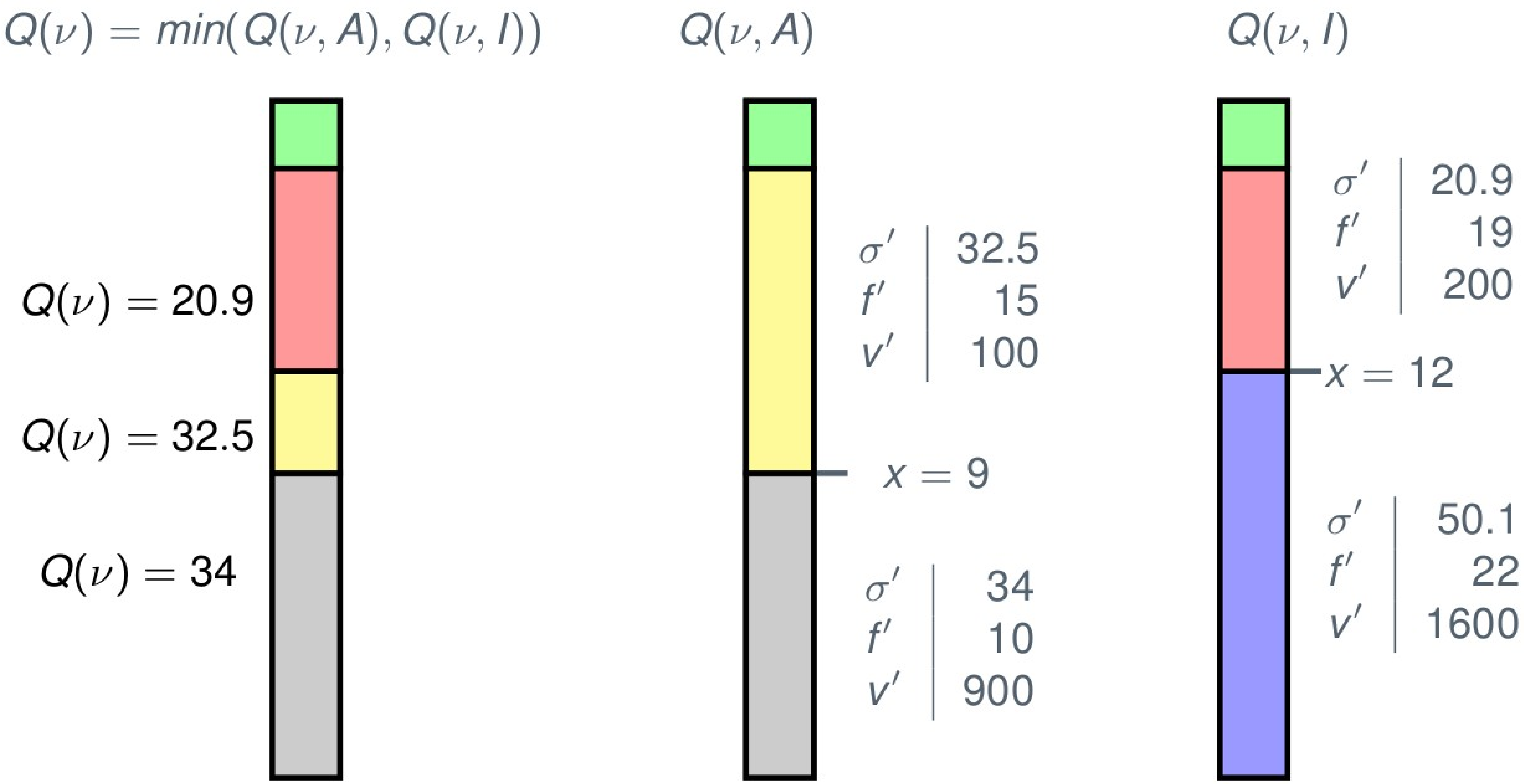


Norm(x,20.9,14.4)
Norm(x,50.1,40)
Welches T-test
Norm(x,34,30)
Norm(x,32.5,10)
Kolmogorow-Smirnoff test
Q(ν, A)
Q(ν, I)
σ′ 32.5
f′ 15
v′ 100
x = 9
σ′ 34
f′ 10
v′ 900
σ′ 20.9
f′ 19
v′ 200
x = 12
Q(ν, I) = 30
σ′ 50.1
f′ 22
v′ 1600

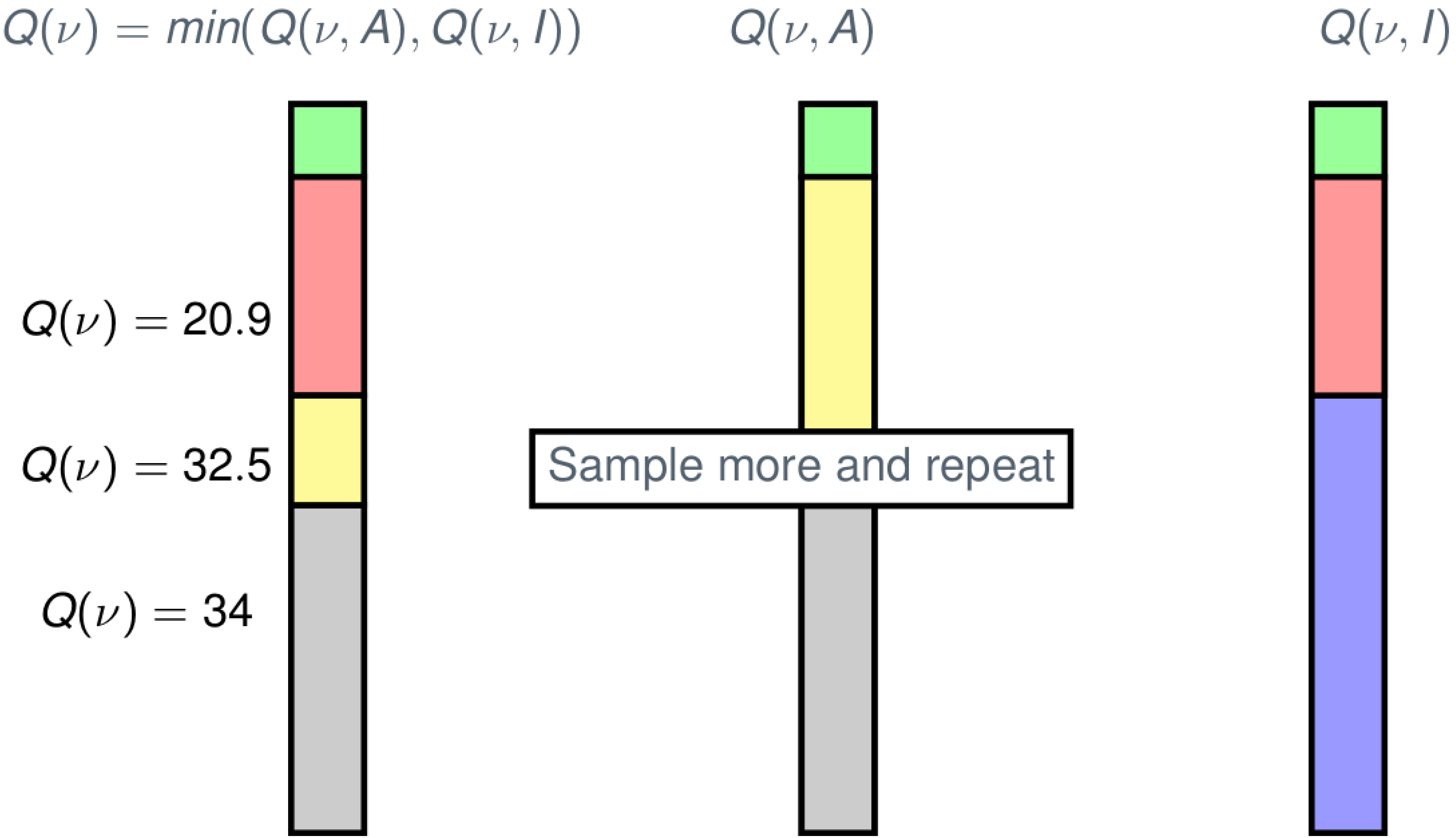

Norm(x,20.9,14.4)
Norm(x,50.1,40)
Welches T-test
Kolmogorow-Smirnoff test
$Q(\nu, A)$
$Q(\nu, I)$
σ' 32.5
f' 15
v' 100
σ' 34
f' 10
v' 900
33
σ' 20.9
f' 19
v' 200
$x = 12$
$Q(\nu, I) = 30$
σ' 50.1
f' 22
v' 1600
Disclaimer: Nonsense statistics; it is a heuristic
Approximated in implementation

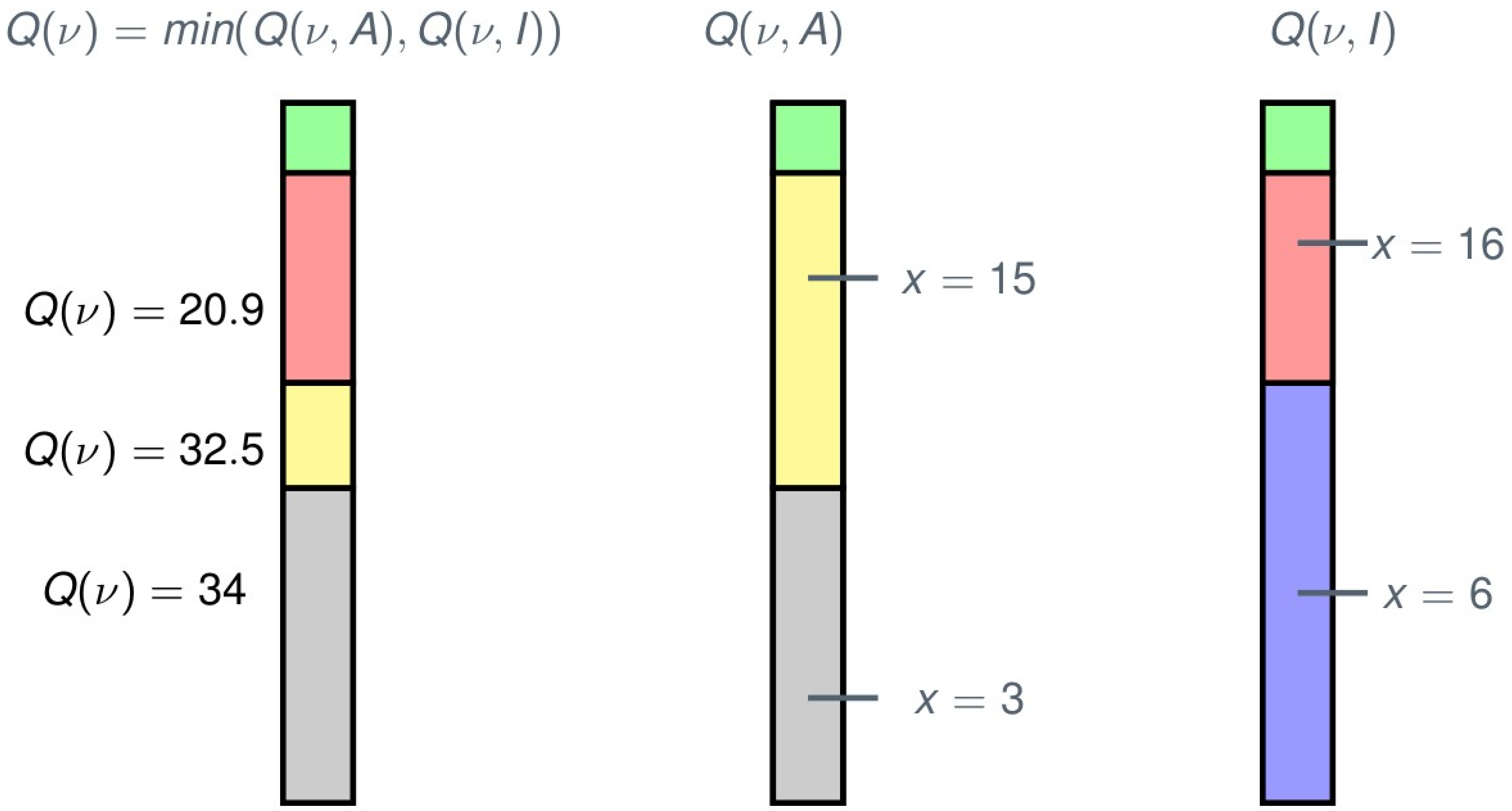


$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu) = 30$
$Q(\nu, A)$
σ' | 32.5
f' | 15
v' | 100
$x = 9$
σ' | 34
f' | 10
v' | 900
$Q(\nu, I)$
σ' | 20.9
f' | 19
v' | 200
$x = 12$
σ' | 50.1
f' | 22
v' | 1600



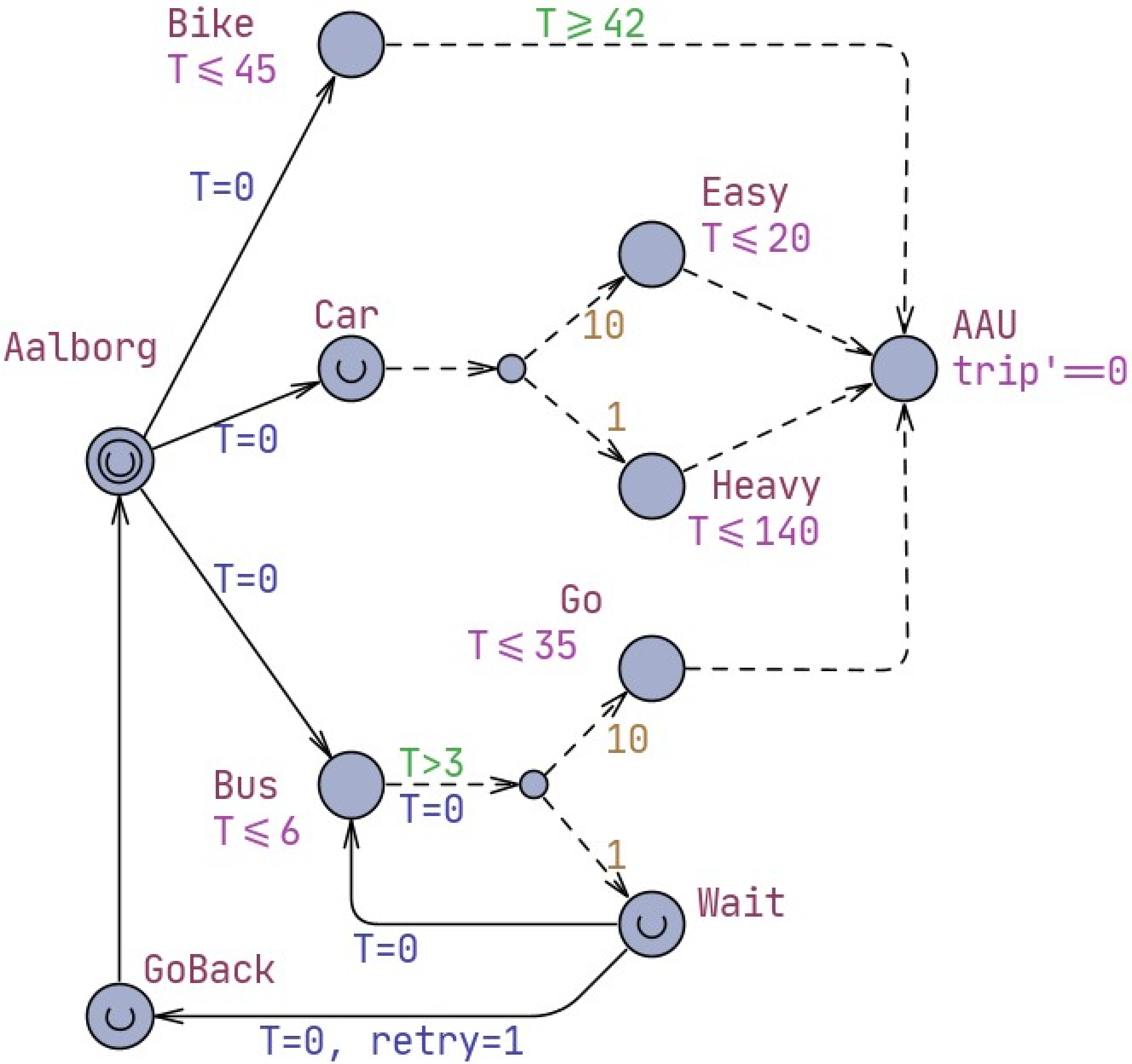
$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$Q(\nu) = 20.9$
$Q(\nu) = 32.5$
$Q(\nu) = 34$
σ' 32.5
f' 15
v' 100
$x = 9$
σ' 34
f' 10
v' 900
σ' 20.9
f' 19
v' 200
$x = 12$
σ' 50.1
f' 22
v' 1600


$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$Q(\nu) = 20.9$
$Q(\nu) = 32.5$
$Q(\nu) = 34$
Sample more and repeat


$Q(\nu) = min(Q(\nu, A), Q(\nu, I))$
$Q(\nu, A)$
$Q(\nu, I)$
$x = 16$
$x = 15$
$Q(\nu) = 20.9$
$Q(\nu) = 32.5$
$Q(\nu) = 34$
$x = 6$
$x = 3$

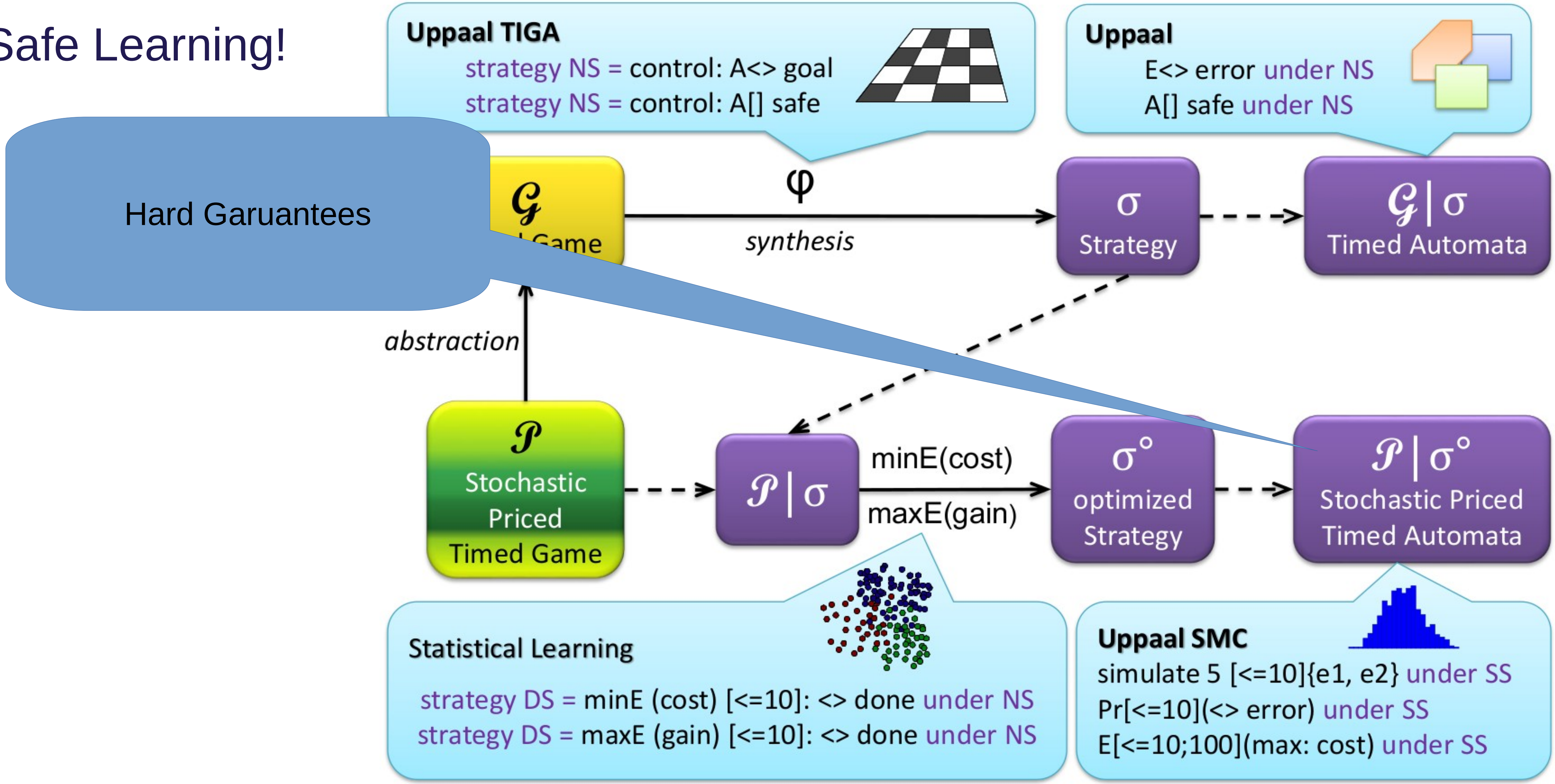
DEMO

Safe Learning!
The curious case of going to work

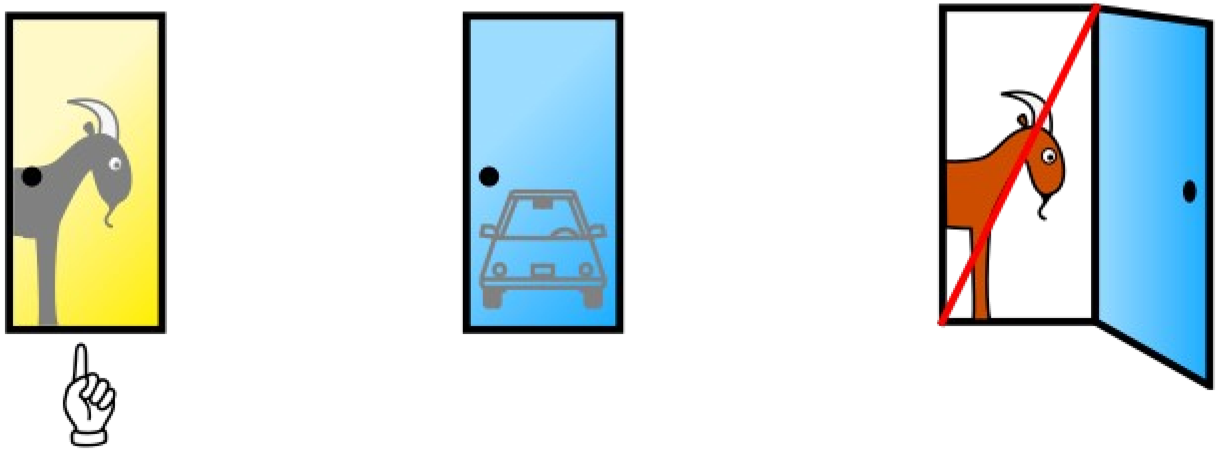


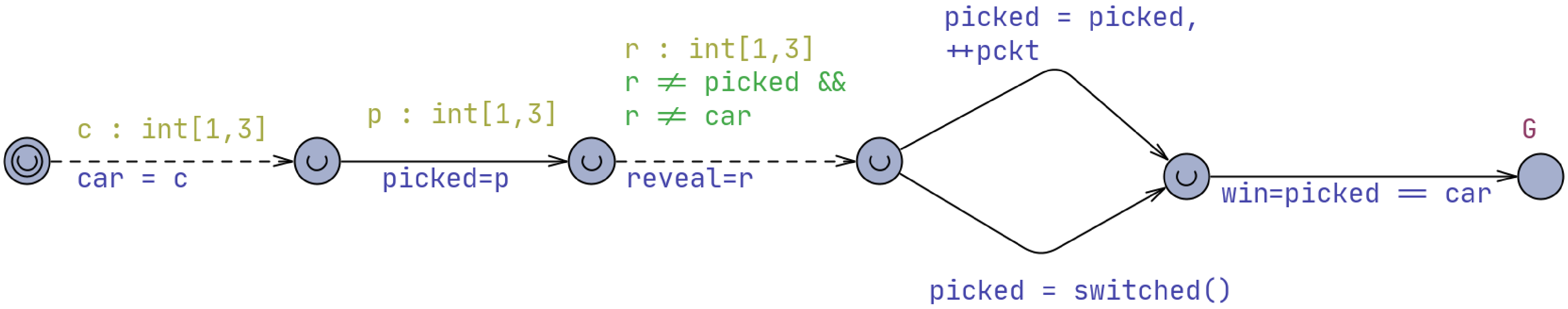
What is the fastest way to get to work?
What if Kim needs to be there in 40 min?

Safe Learning!



Partial Observability in Stratego
The Monty Hall Problem





We cannot observe what is behind the door!

```
strategy s = maxE(win) [<=1] : <> Hall.G // observes everything
strategy s = maxE(win) [<=1] {Hall.location, reveal} -> {}: <> Hall.G
```

Partial Observability in Stratego
The query explained

Timebound
e.g. time<=100

Goal-condition, often
inverse of timebound:
time>=100 || WIN

```
strategy s = maxE(...) [<=...] { ... } -> { ... }: <> ...
```

Cost expression

Variables to be explicit about
I.e. variables with few values
Locations, booleans ...
Defaults to all locations, ints, bools

Variables to generalize over
I.e. variables with many values:
Doubles, clocks ...
Defaults to all clocks & doubles

Tips when using Stratego

Non-lazy controllers

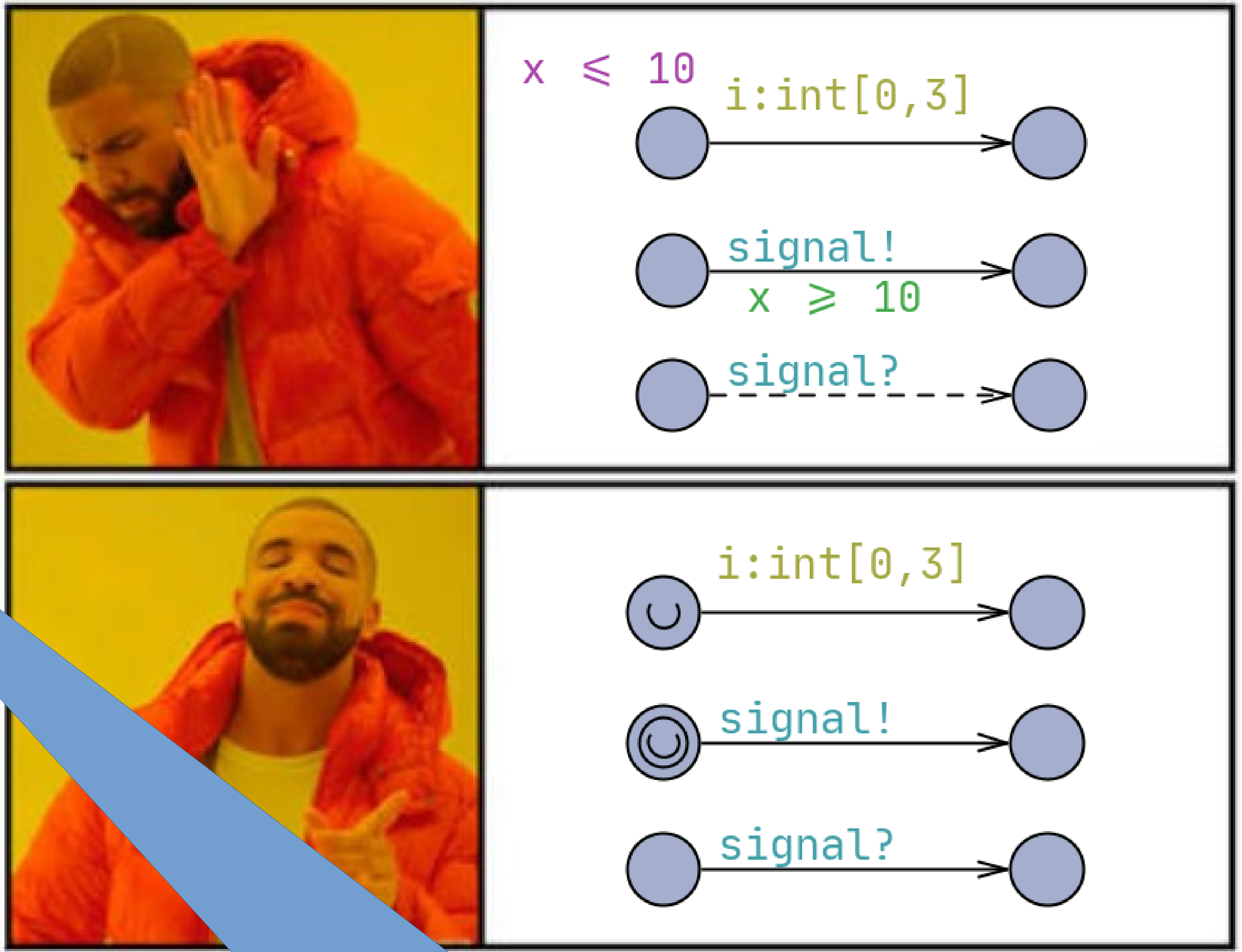
- Act now or “wait forever”
- Stratego cannot propose a delay!
- Keep “controllable locations” urgent
  - Avoid “lazy” behaviour

Problem is Markovian

- Future is independent from past
- Cost-function is Markovian
- The observable projection is Markovian!

Other Tips

- Keep receivers controllable
  - Avoids consistency-issue with TiGa
- Avoid guards on controllable edges
  - Breaks assumption of Q-learning

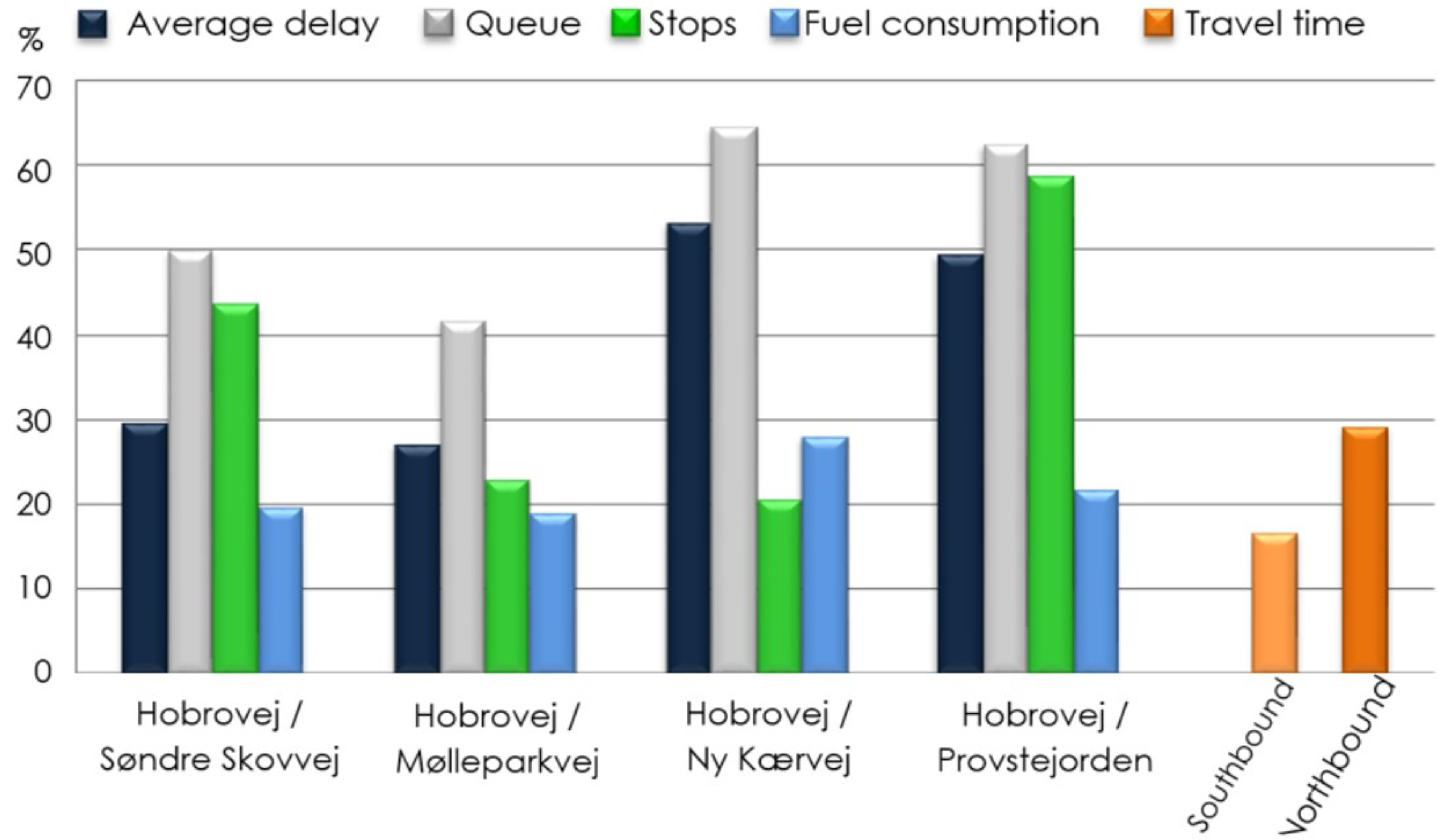


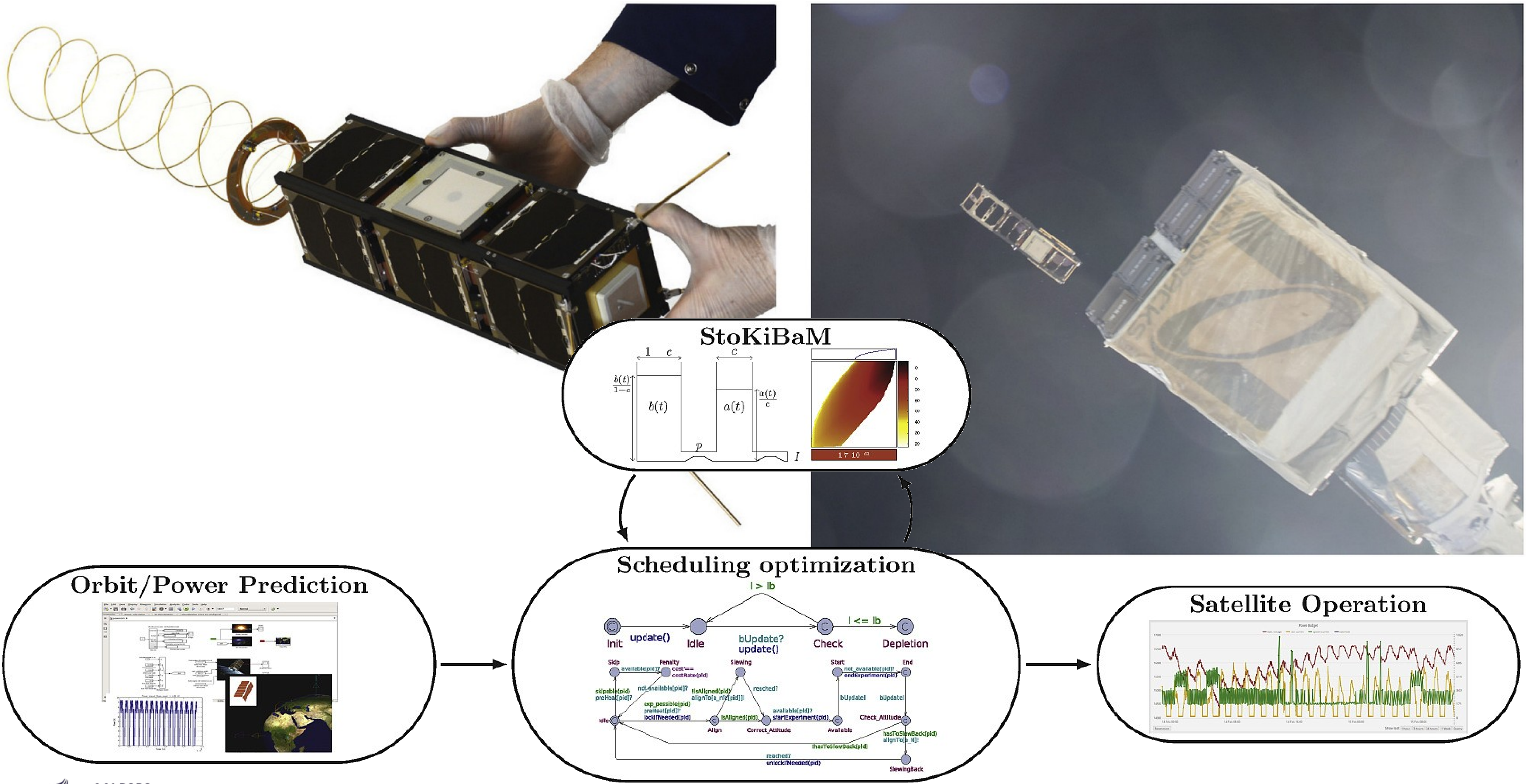
WIP to improve all of these things

Exercise!

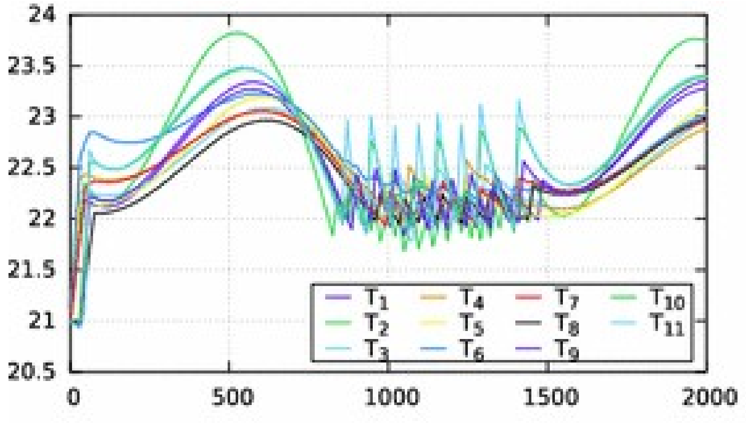
3) Stochastic Jobshop Scheduling Game

- Extend your Stochastic Jobshop scheduling to a stochastic game
- Optimize the scheduling using Stratego, minimize the completion time
- Make a strategy (using TiGa) that garuantees that Kim is done in 60 minutes
  - He has a plane to catch!
- Compare the random scheduler, the fast scheduler and the “Kim gets to the plane”-scheduler
- Try to improve (using Stratego) the “Kim gets to the plane”-strategy
- Change the observations of stratego s.t. only the “sections in use” and the location of the persons are visible, what is the effect?

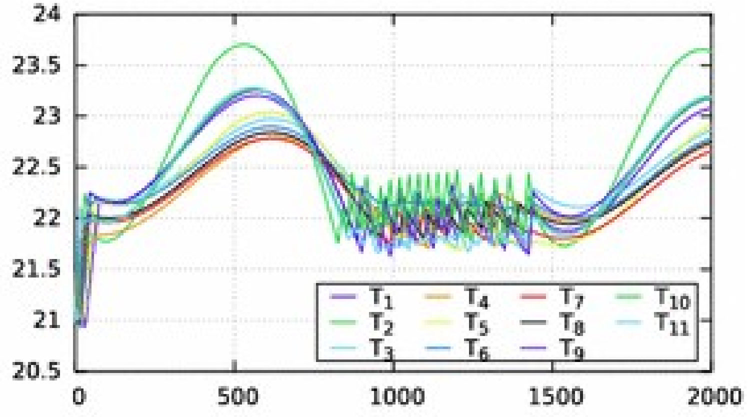

Average delay
Queue
Stops
Fuel consumption
Travel time
%
70
60
50
40
30
20
10
0
Hobrovej / Søndre Skovvej
Hobrovej / Mølleparkvej
Hobrovej / Ny Kærvej
Hobrovej / Provstejorden
Southbound
Northbound


StoKiBaM
Scheduling optimization
I > Ib
I <= Ib
Init
update()
Idle
bUpdate?
update()
Check
Depletion
Orbit/Power Prediction
Satellite Operation

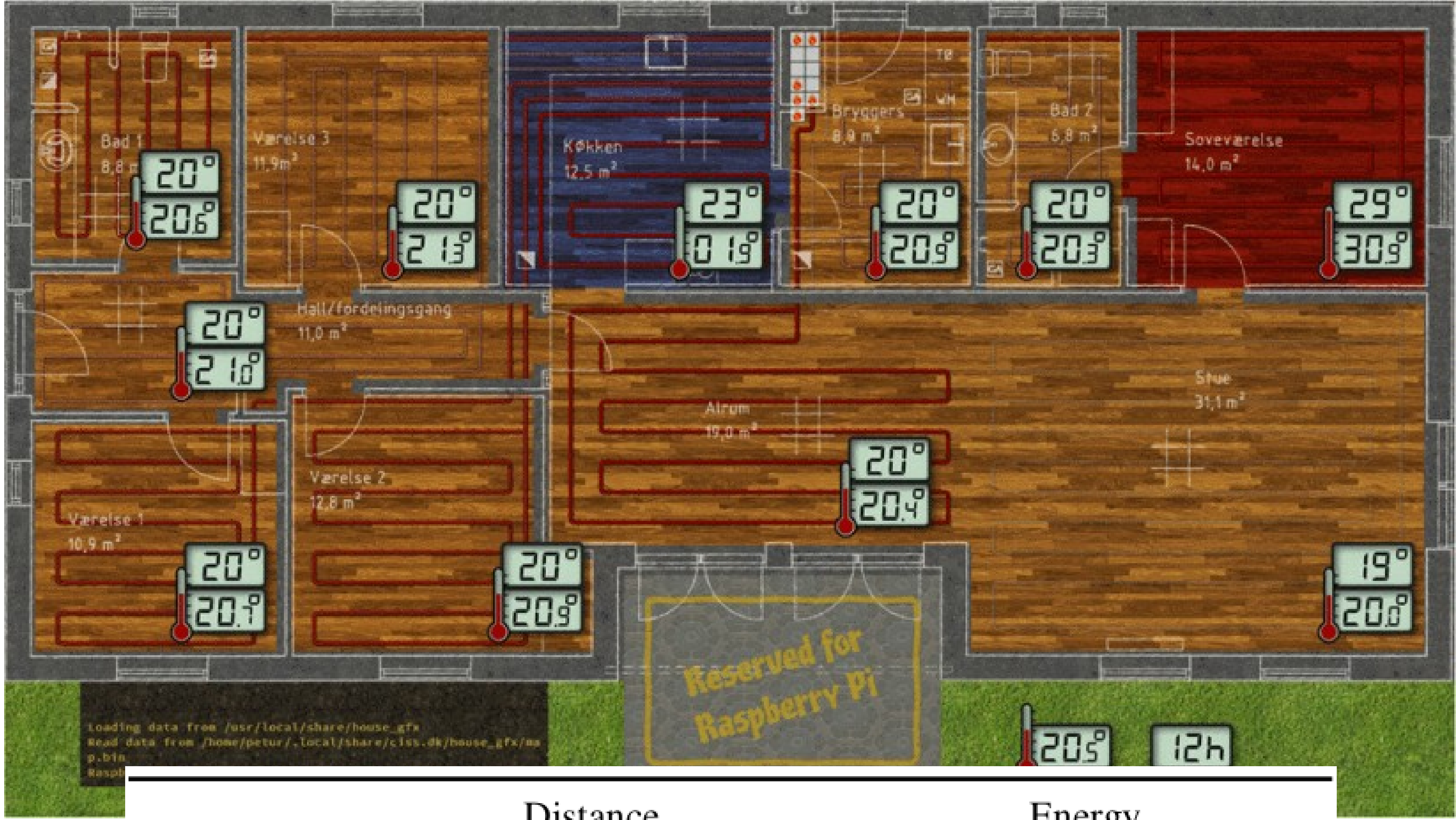
Examples



(a) Bang-Bang controller



(b) STRATEGO-ON-CL controller



Weather	Distance			Energy		
	Bang-Bang	STRATEGO	imp.	Bang-Bang	STRATEGO	imp.
Aalborg	14583	8342	43%	14180	12626	10%
Anadyr	2385515	1483272	37%	23040	22475	2%
Ankara	17985	10464	41%	17468	15684	10%
Minneapolis	22052	12175	44%	18165	15882	12%
Murmansk	399421	187941	52%	22355	21011	6%

Things we have not discussed

- Differential equations
- Loading/saving strategies
- Interfacing with external C-code
- Loading real-life data
- On-line reinforcement learning
  - Adaptive/data-driven planning