Contributions on Solving COVID-19 Crisis with Reinforcement Learning

Presented by Francesco Mikulis-Borsoi & Tonmoay Deb

May 2021

Implementing SR-MILP in Python

- Team Goal: Generate Dataset and Implement the SR-MILP to check if the output is consistent with the RCraam code.

Implementing SR-MILP in Python

- Team Goal: Generate Dataset and Implement the SR-MILP to check if the output is consistent with the RCraam code.
- Number of states: 10, Number of action: 2, Number of outcomes: 10

Implementing SR-MILP in Python

- Team Goal: Generate Dataset and Implement the SR-MILP to check if the output is consistent with the RCraam code.
- Number of states: 10, Number of action: 2, Number of outcomes: 10
- State formulation:

% infected (of susceptible)	State
$[0, 1]$	0
$[1, 2]$	1
$[2, 3]$	2
$[3, 4]$	3
$[4, 5]$	4
$[5, 6]$	5
$[6, 7]$	6
$[7, 8]$	7
$[8, 9]$	8
$[9, 100]$	9

Implementing SR-MILP in Python

- Actions (policy): $0 \rightarrow$ OPEN; $1 \rightarrow$ CLOSE
- The reward formulation strategy:
  - if closed and $< 5\%$ sick: -0.5 # don't close pre-emptively (economy?)

Implementing SR-MILP in Python

- Actions (policy): $0 \rightarrow$ OPEN; $1 \rightarrow$ CLOSE
- The reward formulation strategy:
  - if closed and $< 5\%$ sick: -0.5 # don't close pre-emptively (economy?)
  - if closed and $\geq 5\%$ sick: 0.0 # close if many people are sick!

Implementing SR-MILP in Python

- Actions (policy): $0 \rightarrow$ OPEN; $1 \rightarrow$ CLOSE
- The reward formulation strategy:
  - if closed and $< 5\%$ sick: -0.5 # don't close pre-emptively (economy?)
  - if closed and $\geq 5\%$ sick: 0.0 # close if many people are sick!
  - if open and $5 \geq \%$ sick: -0.5 # don't keep open if many people are sick.

Implementing SR-MILP in Python

- Actions (policy): $0 \rightarrow$ OPEN; $1 \rightarrow$ CLOSE
- The reward formulation strategy:
  - if closed and $< 5\%$ sick: -0.5 # don't close pre-emptively (economy?)
  - if closed and $\geq 5\%$ sick: 0.0 # close if many people are sick!
  - if open and $5 \geq \%$ sick: -0.5 # don't keep open if many people are sick.
  - if open and 5% sick: depending on how much the number of cases increased, *reward* $\in [0, -0.4]$

Implementing SR-MILP in Python

- Actions (policy): 0 $\rightarrow$ OPEN; 1 $\rightarrow$ CLOSE
- The reward formulation strategy:
  - if closed and $< 5\%$ sick: -0.5 $\#$ don't close pre-emptively (economy?)
  - if closed and $\geq 5\%$ sick: 0.0 $\#$ close if many people are sick!
  - if open and $5 \geq \%$ sick: -0.5 $\#$ don't keep open if many people are sick.
  - if open and 5% sick: depending on how much the number of cases increased, *reward* $\in [0, -0.4]$
- Outcome: It worked! It found the optimal policy as it did in the Rcraam code. Also, it adaps to multiple outcomes now.

DataSet Table

Table: Table used for the input to the SR-MILP

idstatefrom	idaction	idstateto	probability	reward	idoutcome
0	0	0	0.5	-0.5	0
0	0	1	0.5	-0.5	0
0	0	0	0.1	-0.5	1
0	0	1	0.1	-0.5	1
0	0	0	1	-0.5	2
0	0	1	0	-0.5	2
0	0	0	1	-0.5	3
0	0	1	0	-0.5	3
0	0	0	0.1	-0.5	4
0	0	1	0.1	-0.5	4
0	0	0	1	-0.5	5
0	0	1	0	-0.5	5

Dataset Table (Continued)

Table: Table used for the input to the SR-MILP

idstatefrom	idaction	idstateto	probability	reward	idoutcome
9	1	8	0.058824	-0.5	0
9	1	9	0.941176	-0.5	0
9	1	8	0	-0.5	1
9	1	9	1	-0.5	1
9	1	8	0	-0.5	2
9	1	9	0.962963	-0.5	2
9	1	8	0	-0.5	3
9	1	9	1	-0.5	3
9	1	8	0	-0.5	4
9	1	9	1	-0.5	4
9	1	8	0	-0.5	5
9	1	9	1	-0.5	5

The overall SR-MILP Formulation [LGP20]

$$
\begin{array}{lll}
\underset{\substack{\pi\in\{0,1\}^{S\times A},\, b\in\mathbb{R},\\ u\in\mathbb{R}_+^{S\times A\times N},\, y\in\mathbb{R}_+^N}}{\text{maximize}} & \lambda\cdot\left(b-\frac{1}{1-\alpha}\sum_{\omega\in\Omega}y(\omega)\right)+(1-\lambda)\cdot\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}}\sum_{\omega\in\Omega}u(s,a,\omega)\sum_{s'\in\mathcal{S}}r(s,a,s')\cdot P^{\omega}(s,a,s') & \\
\text{subject to} & y(\omega)-b\cdot f_\omega \;\geq\; -\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}}u(s,a,\omega)\sum_{s'\in\mathcal{S}}P^{\omega}(s,a,s')\cdot r(s,a,s'), & \omega\in\Omega, \\
& \sum_{a\in\mathcal{A}}u(s,a,\omega) \;=\; \sum_{s'\in\mathcal{S}}\sum_{a'\in\mathcal{A}}\gamma\cdot u(s',a',\omega)\cdot P^{\omega}(s',a',s)+f_\omega\cdot p_0(s), & s\in\mathcal{S},\ \omega\in\Omega, \\
& \sum_{a\in\mathcal{A}}\pi(s,a) \;=\; 1, & s\in\mathcal{S}, \\
& u(s,a,\omega) \;\leq\; f_\omega\cdot\pi(s,a)/(1-\gamma), & s\in\mathcal{S},\ a\in\mathcal{A},\ \omega\in\Omega.
\end{array}
$$

Figure: Overview of SR-MILP approach we applied on *COVID MDP*

.

Discovered policies using SR-MILP

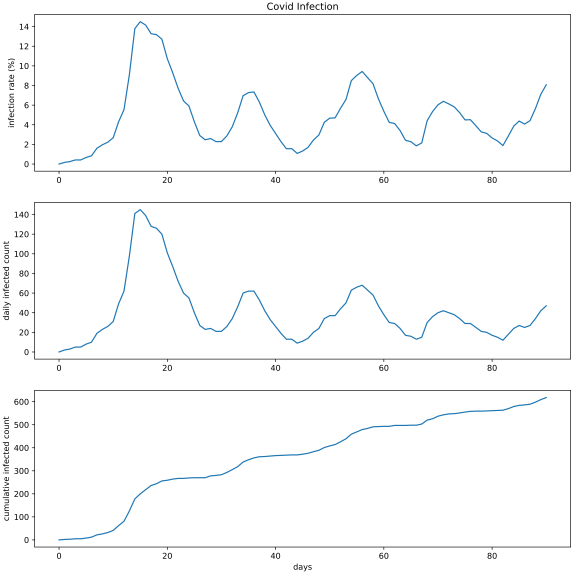
State	Policy
0	OPEN
1	OPEN
2	OPEN
3	CLOSE
4	OPEN
5	CLOSE
6	CLOSE
7	CLOSE
8	CLOSE
9	CLOSE

Table: SR-MILP robust policy

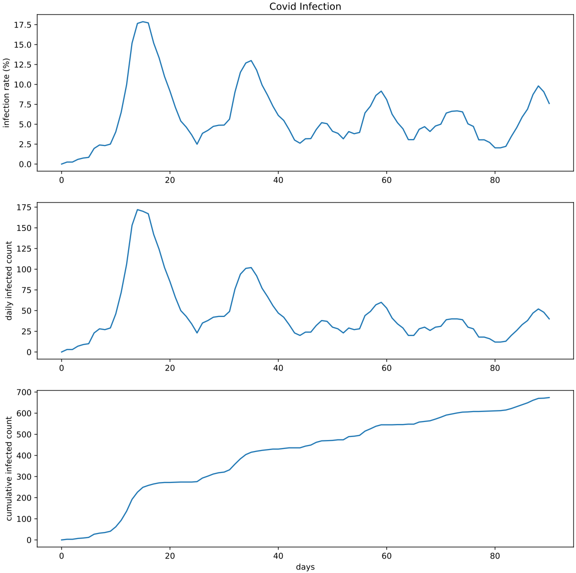
State	Policy
0	OPEN
1	OPEN
2	OPEN
3	OPEN
4	OPEN
5	CLOSE
6	CLOSE
7	CLOSE
8	CLOSE
9	CLOSE

Table: SR-MILP CRAAM robust policy

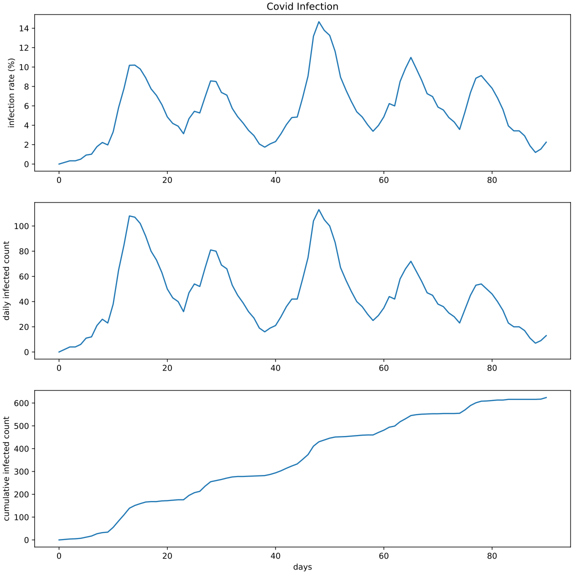
Our SR-MILP implementation - 1



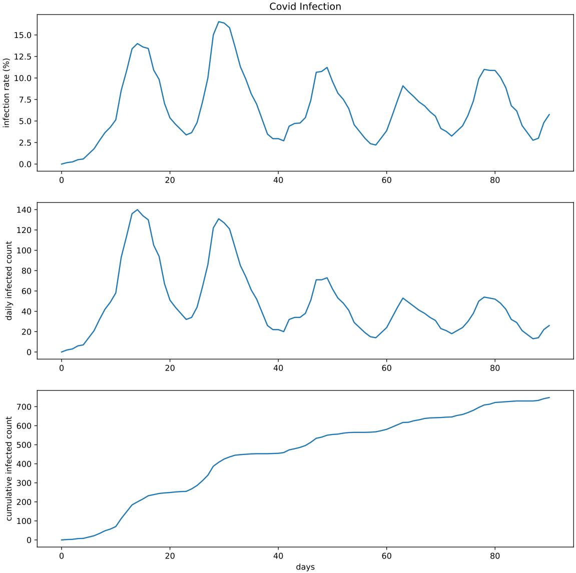
Our SR-MILP implementation - 3



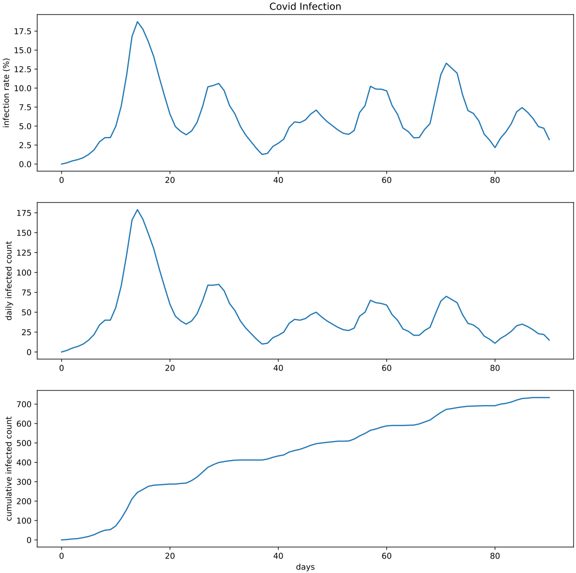
Our SR-MILP implementation - 3



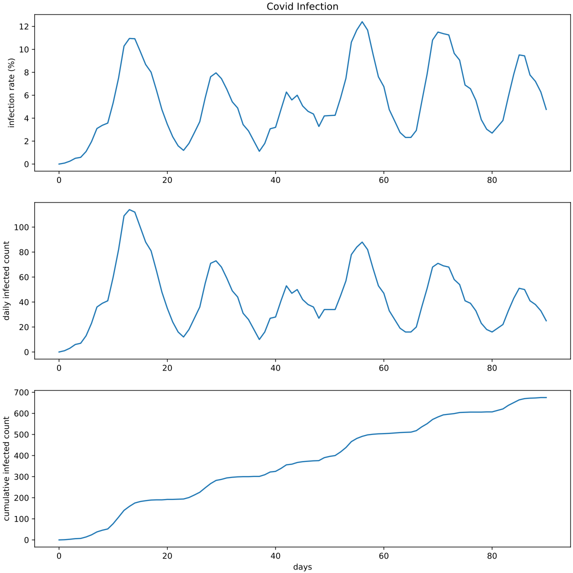
CRAAM SR-MILP implementation - 1



CRAAM SR-MILP implementation - 3



CRAAM SR-MILP implementation - 3



References

India's daily covid-19 death toll hits record high, https://www.wsj.com/livecoverage/covid-2021-04-30, Accessed: 2021-5-3.

Elita A Lobo, Mohammad Ghavamzadeh, and Marek Petrik, *Soft-Robust algorithms for batch reinforcement learning*.

Wikipedia contributors, *COVID-19*, https://en.wikipedia.org/w/index.php?title=COVID-19&oldid=1021198556, May 2021, Accessed: 2021-5-3.