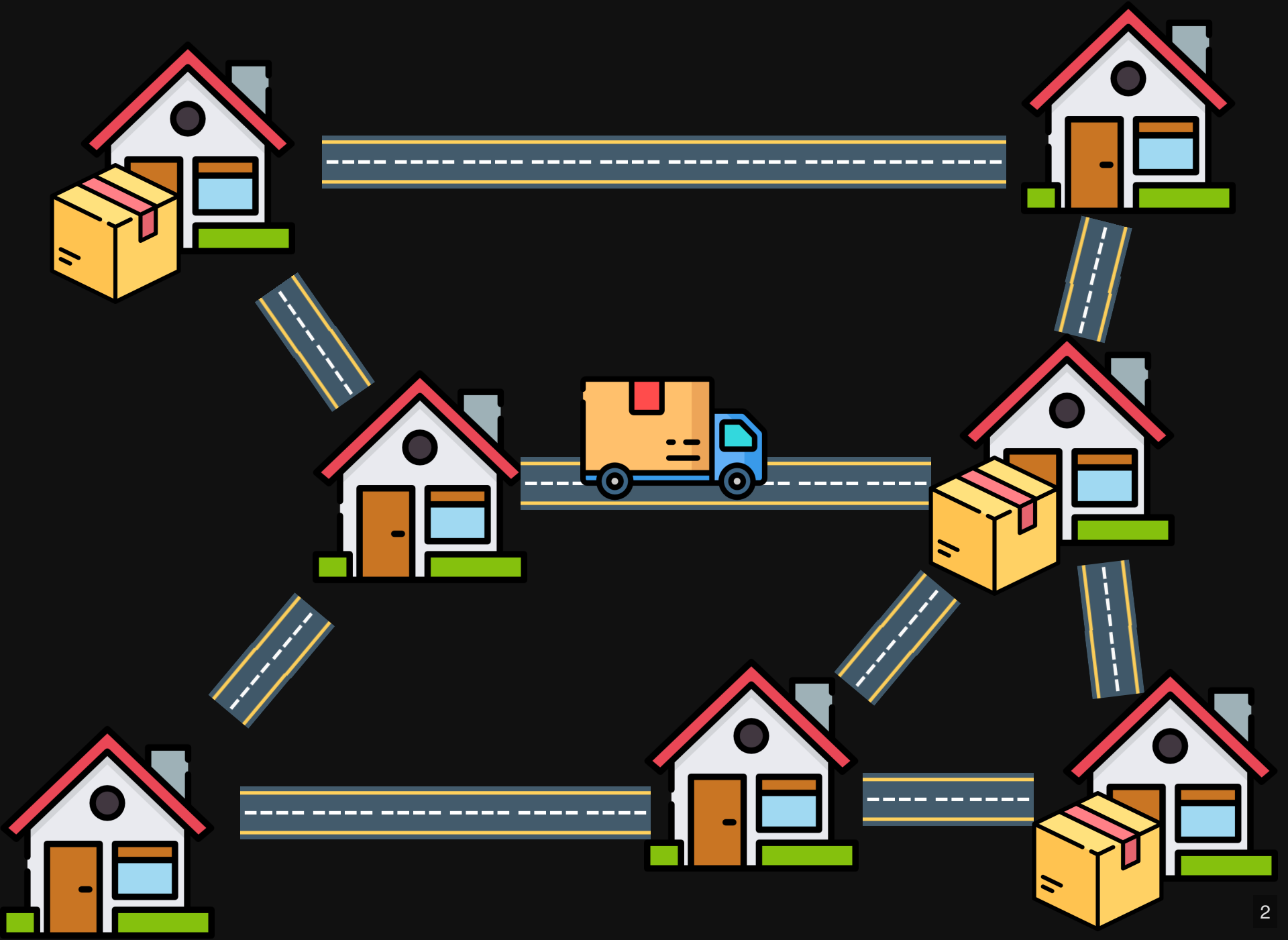
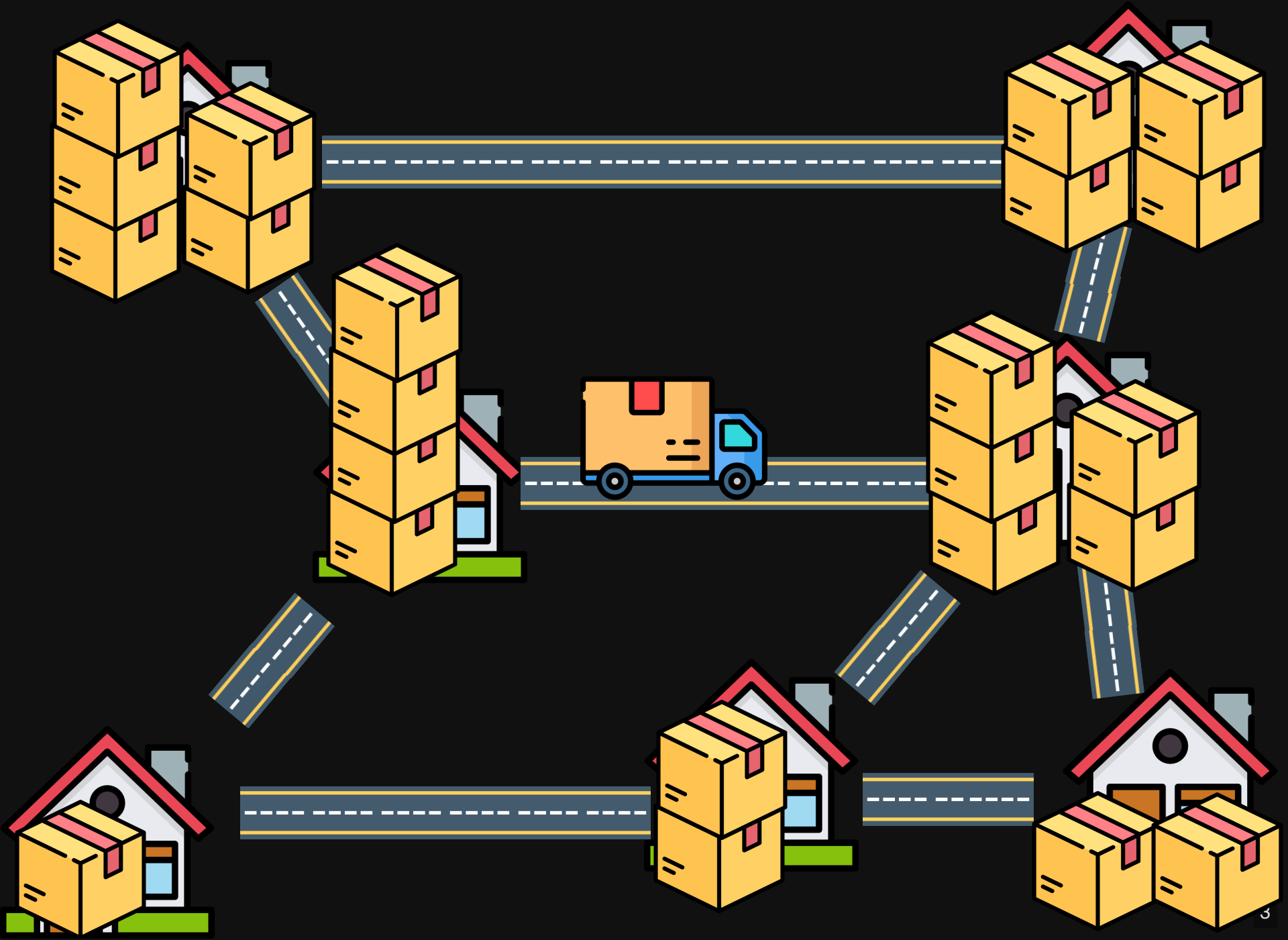
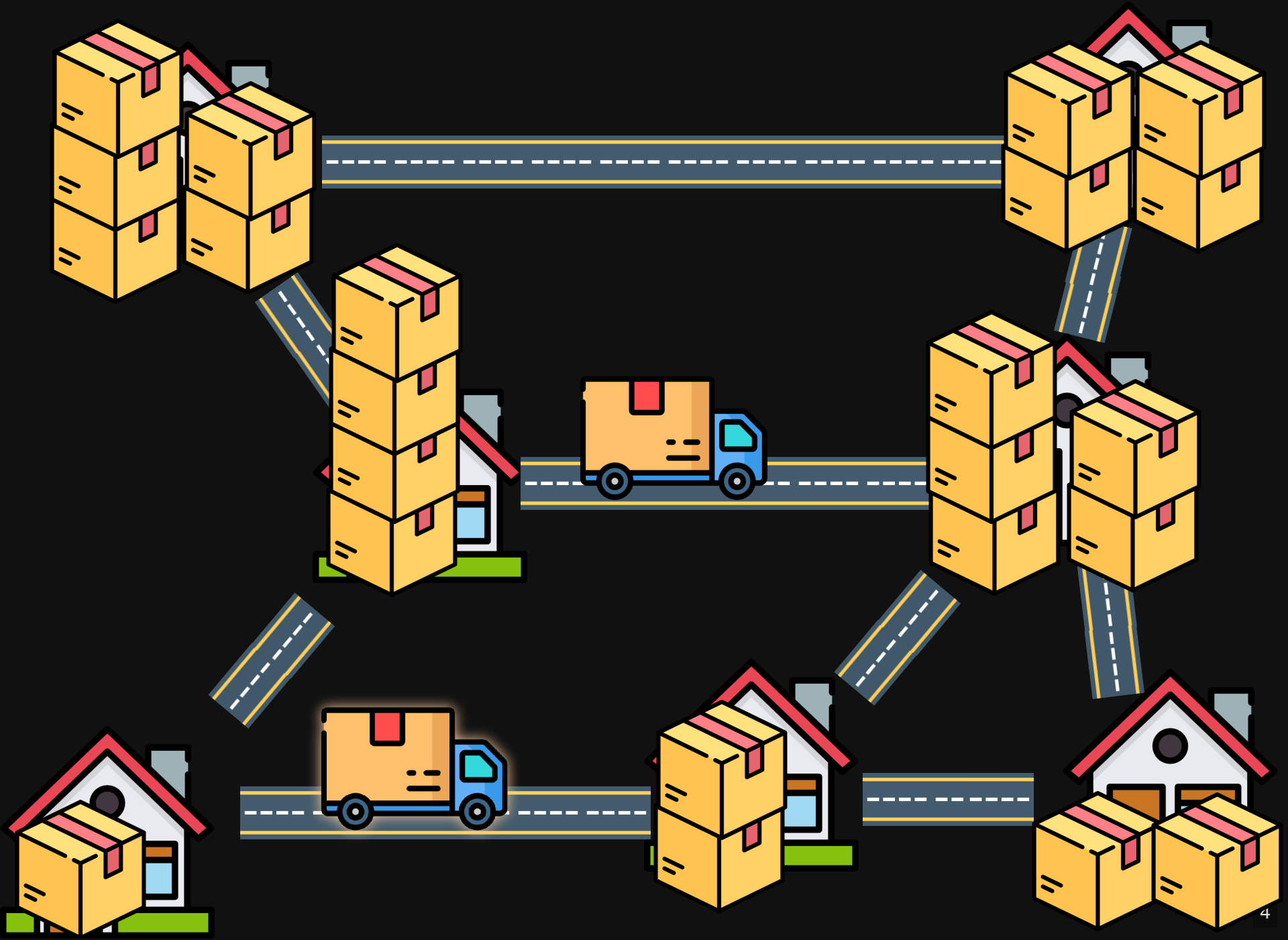
Planning with Object Creation

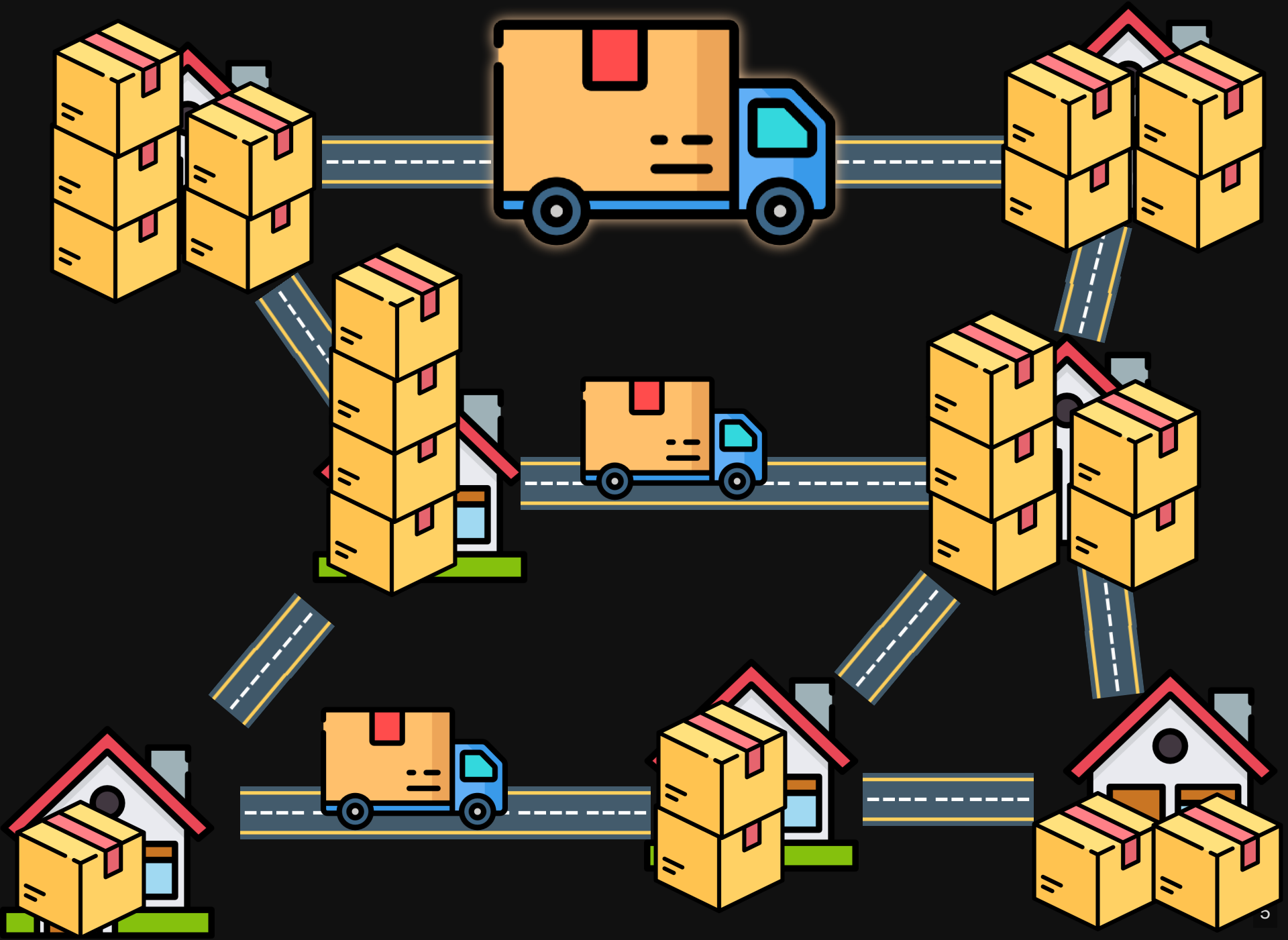
Augusto B. Corrêa, Giuseppe De Giacomo,
Malte Helmert, and Sasha Rubin

University of Basel, Switzerland,
University of Oxford, UK
University of Sydney, Australia









Previous Work

Hoffmann et al. (2009):
web service composition (specific case)

Fuentetaja & de la Rosa (2016):
compile irrelevant objects to counters

Edelkamp et al. (2019):
compile to model-checking

Our Work

synergy between object creation and lifted heuristic search

complete semantics of classical planning with object creation in action effects

PDDL Snippet

```
(:action buy-largest-possible-truck
(:parameters (?C - city)
(:precondition (has-garage ?C)
(:effect (and (when (large-garage ?C)
                    (:new (?T - large-truck)
                          (at ?T ?C)))
              (when (not (large-garage ?C))
                    (:new (?T - small-truck)
                          (at ?T ?C)))))
```

How to plan?

if task is solvable:
lifted breadth-first search finds a plan!

otherwise:
search might go on forever

planning with object creation
is semidecidable

How to search?

```
1  def bfs(task):
2      queue = Queue(task.initial_state)
3      visited = set()
4      while not queue.empty():
5          s = queue.pop()
6          if is_goal(succ):
7              return extract_plan(succ)
8          for a in s.get_applicable_actions():
9              succ = get_successor(s, a)
10             if succ not in visited:
11                 queue.add(succ)
12                 visited.add(succ)
13     return UNSOLVABLE
```

Applicable Actions

state = database

precondition = CSP/query

if objects change, it does not matter:
new CSP at every state anyway

[Francès 2017; ABC et al. 2020; Horčík & Fišer 2021; Ståhlberg 2023]

How to search?

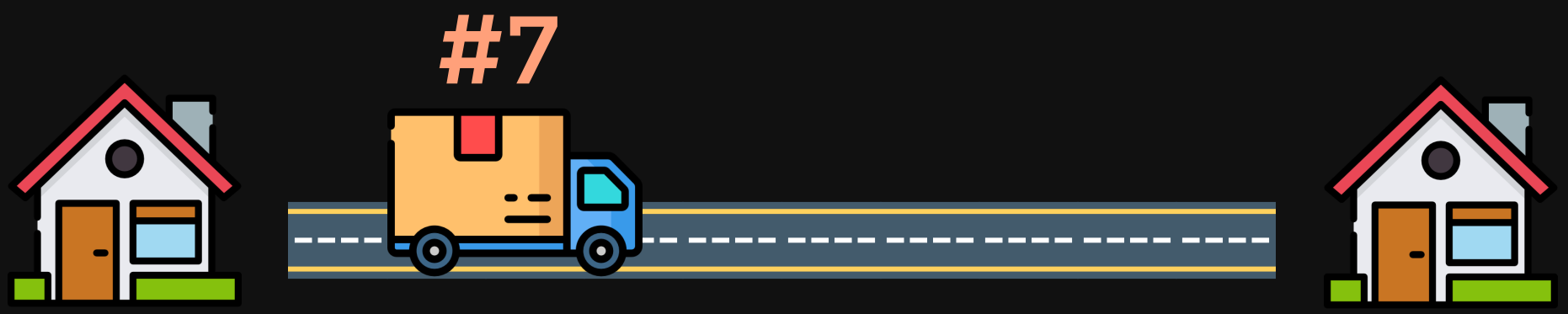
```
 1 def bfs(task):
 2     queue = Queue(task.initial_state)
 3     visited = set()
 4     while not queue.empty():
 5         s = queue.pop()
 6         if is_goal(succ):
 7             return extract_plan(succ)
 8         for a in s.get_applicable_actions()
 9             succ = get_successor(s, a)
10             if succ not in visited:
11                 queue.add(succ)
12                 visited.add(succ)
13     return UNSOLVABLE
```

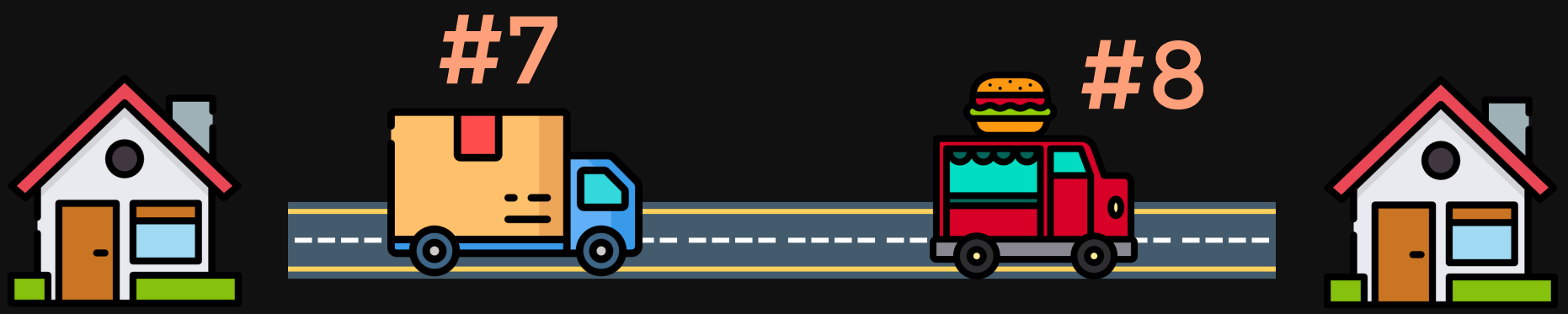
Generate Successors

effects evaluated as in classical planning
except when we have object creation

main difference:

- assign names to created objects
- different ways: numbers, strings, etc.

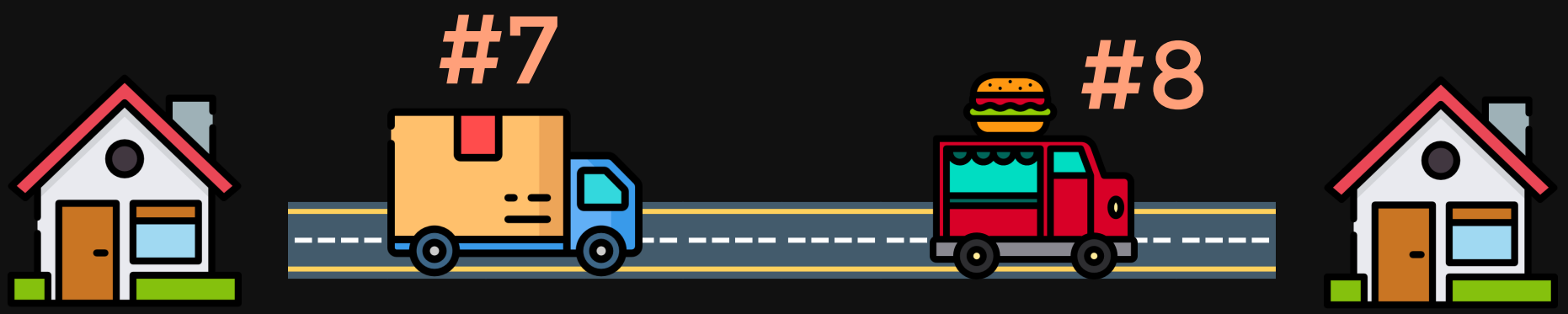

#7


#7
#8

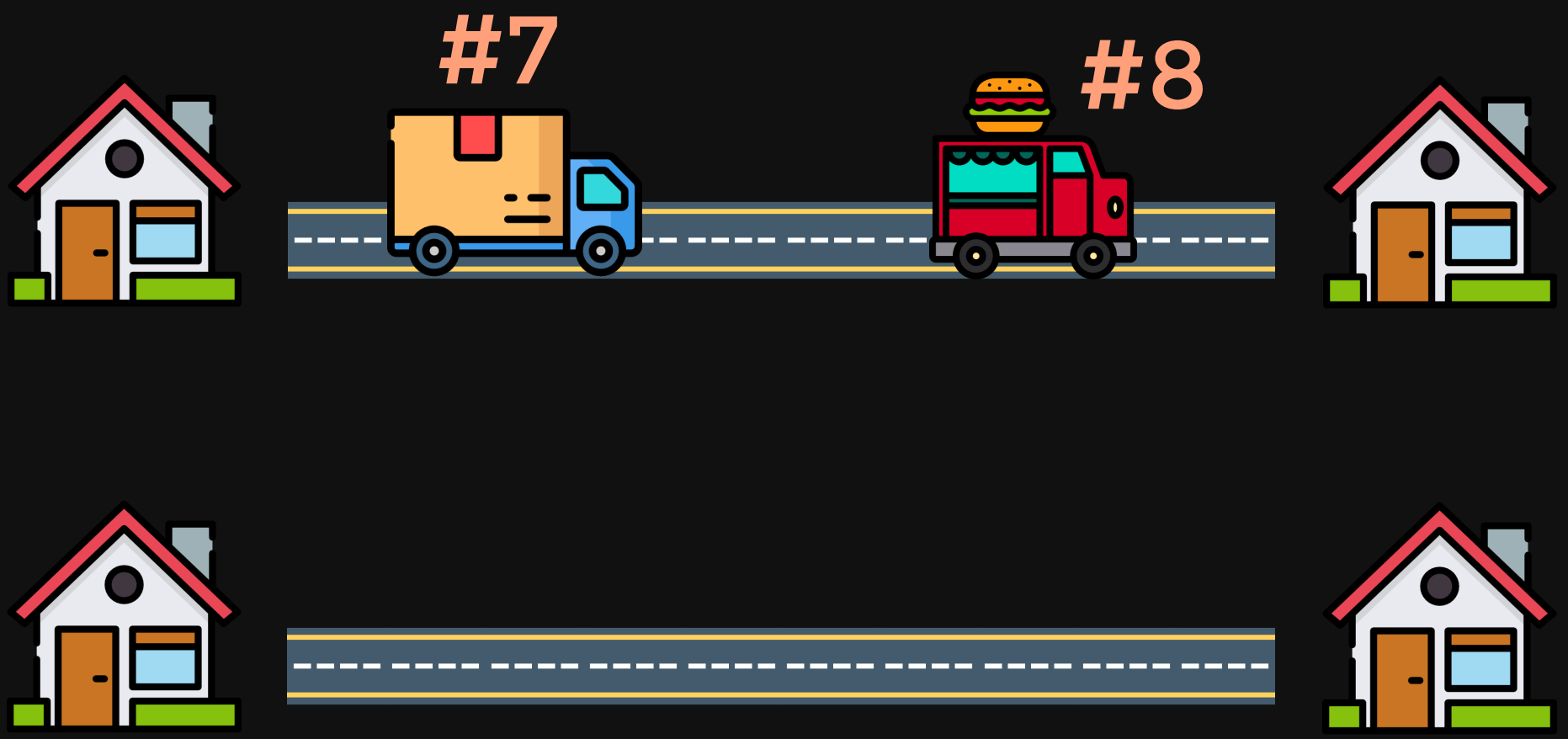
How to search?

```
 1 def bfs(task):
 2     queue = Queue(task.initial_state)
 3     visited = set()
 4     while not queue.empty():
 5         s = queue.pop()
 6         if is_goal(succ):
 7             return extract_plan(succ)
 8         for a in s.get_applicable_actions()
 9             succ = get_successor(s, a)
10             if succ not in visited:
11                 queue.add(succ)
12                 visited.add(succ)
13     return UNSOLVABLE
```

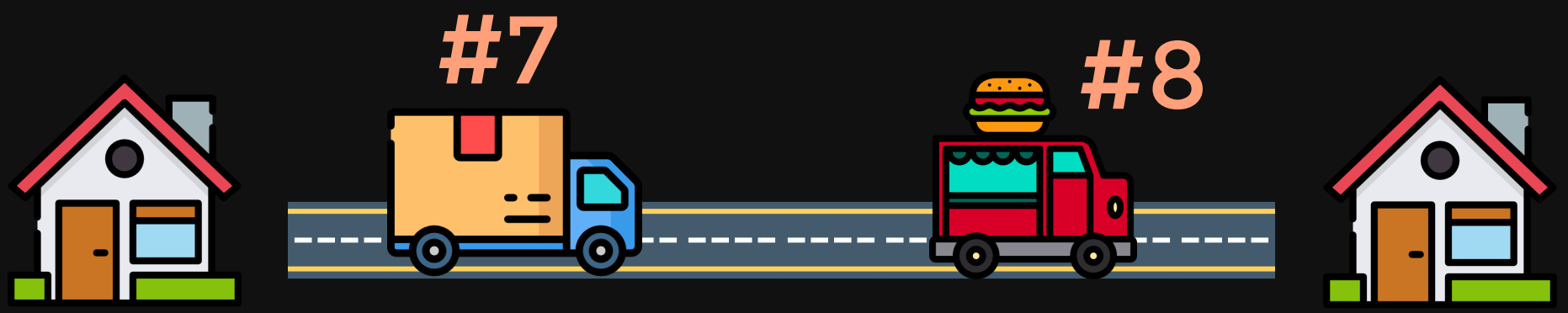
Duplicated States

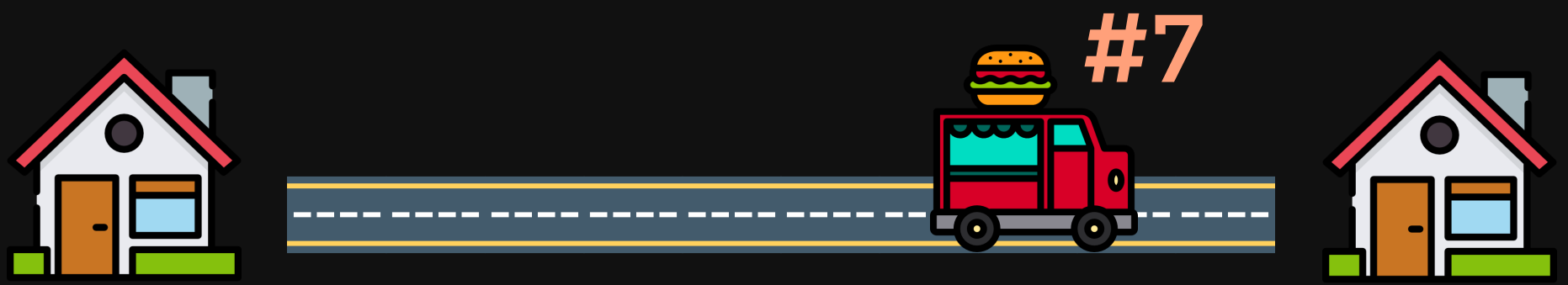


Duplicated States

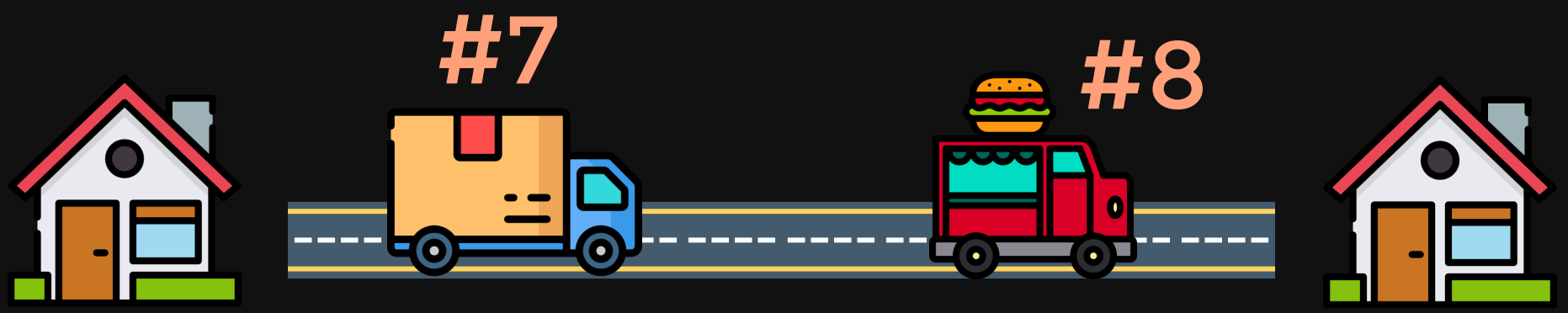


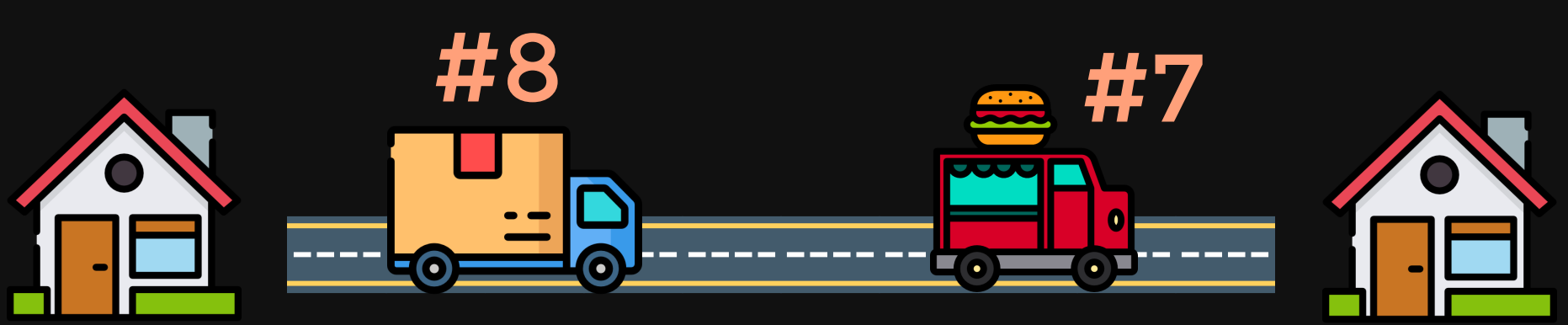
Duplicated States





Duplicated States





Duplicated States

not enough to check equality

need to check state isomorphism

similar to ideas in orbit search

our implementation:
simple duplicate detection

Experimental Results

implemented on top of Powerlifted

four STRIPS-like domains:

1. logistics company
2. cluster management
3. commutative rings (Petrov & Muise 2023)
4. settlers (Long & Fox 2003, IPC 2002)

Experimental Results

model problems simulating object creation:
objects pre-declared; extra predicates

two baselines without object creation:
Powerlifted & Fast Downward (FD)

Breadth-First Search

	w/ creation	no creation
Logistics Comp.	3	5
Cluster Manag.	3	2
Comm. Rings	2	9
Settlers	3	3
Total	**11**	**19**

Problems

3/4 domains have unrestricted creation

branching factor increases at each layer

settlers:
only domain where this is not the case,
and all methods perform similarly

Best-First Width Search

compute novelty and run BFWS

[Lipovetzky & Geffner 2012, 2014, 2017]

every time an object is created,
the successor state has novel tuples!

solution:
partition on # of new objects

BFWS

	w/ creation	no creation
Logistics Comp.	18	8
Cluster Manag.	10	14
Comm. Rings	10	14
Settlers	8	6
Total	**46**	**42**

BFWS

	w/ creation	FD
Logistics Comp.	18	6
Cluster Manag.	10	12
Comm. Rings	10	10
Settlers	8	4
Total	**46**	**32**

Conclusion

object creation in action effects

implemented on top of a lifted planner

a long way to go...

Conclusion

object creation in action effects

implemented on top of a lifted planner

a long way to go...

Thank you!

Breadth-First Search

	w/ creation	FD
Logistics Comp.	3	5
Cluster Manag.	3	5
Comm. Rings	2	8
Settlers	3	3
Total	**11**	**21**