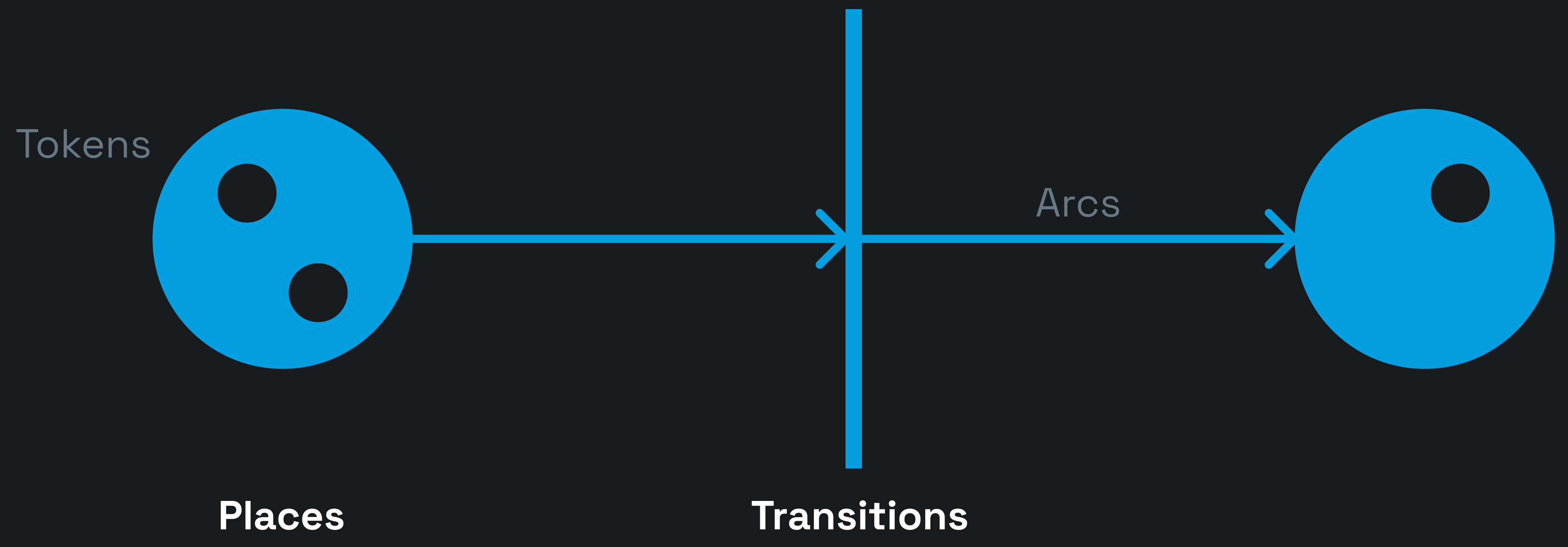
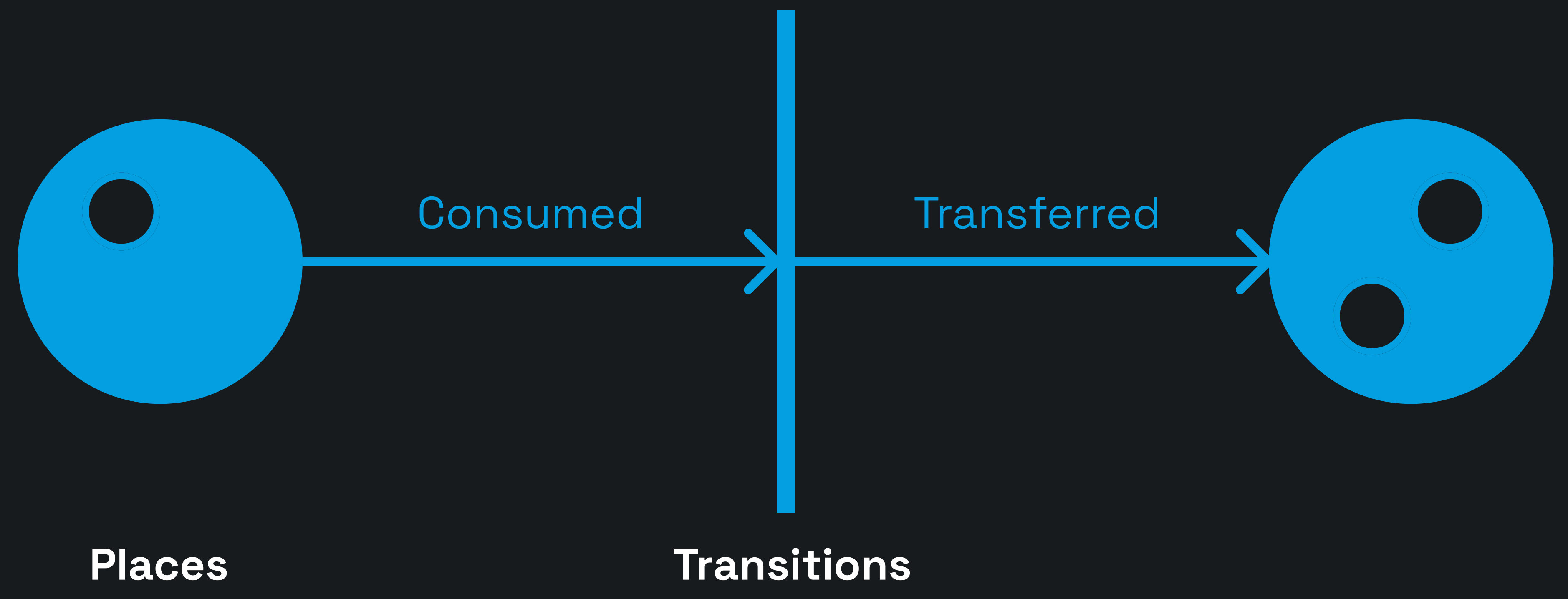
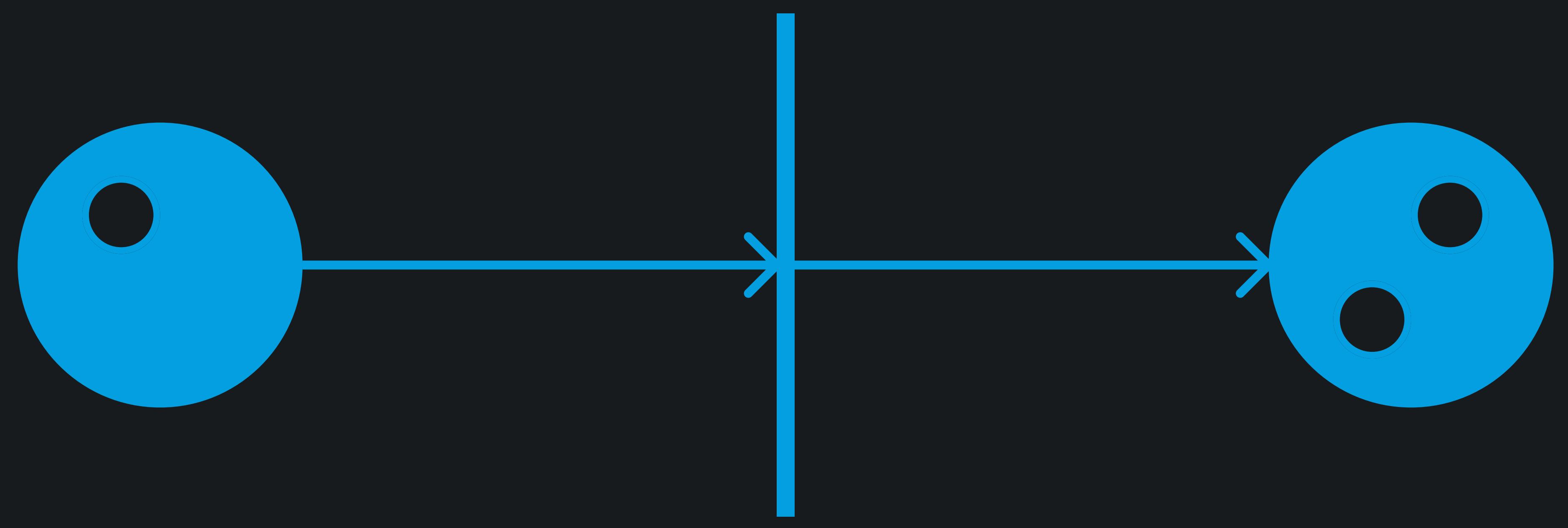
Fighting deadlocks with (modified) **petri nets**


Tokens
Arcs
Places
Transitions

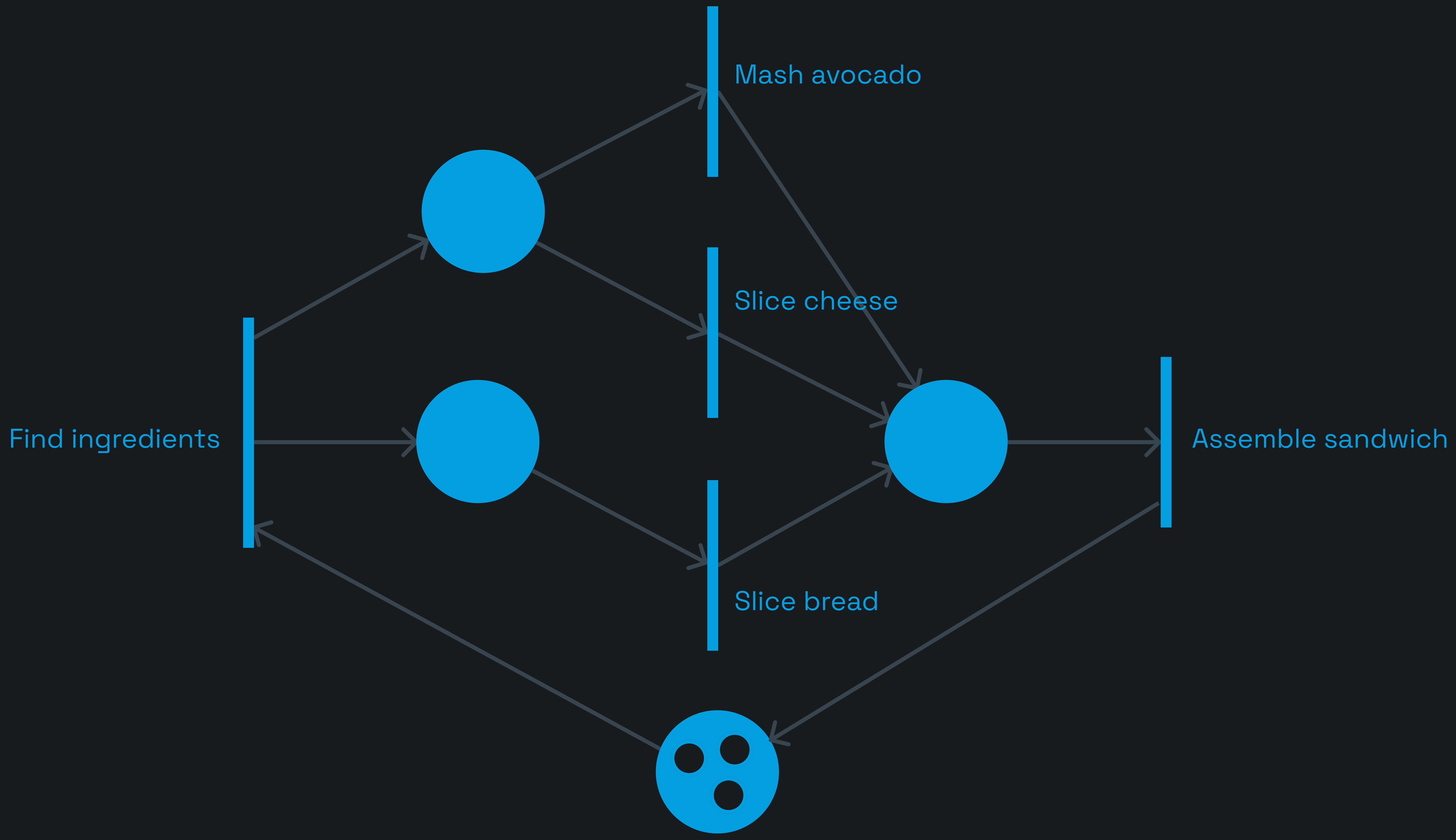

Consumed
Transferred
Places
Transitions

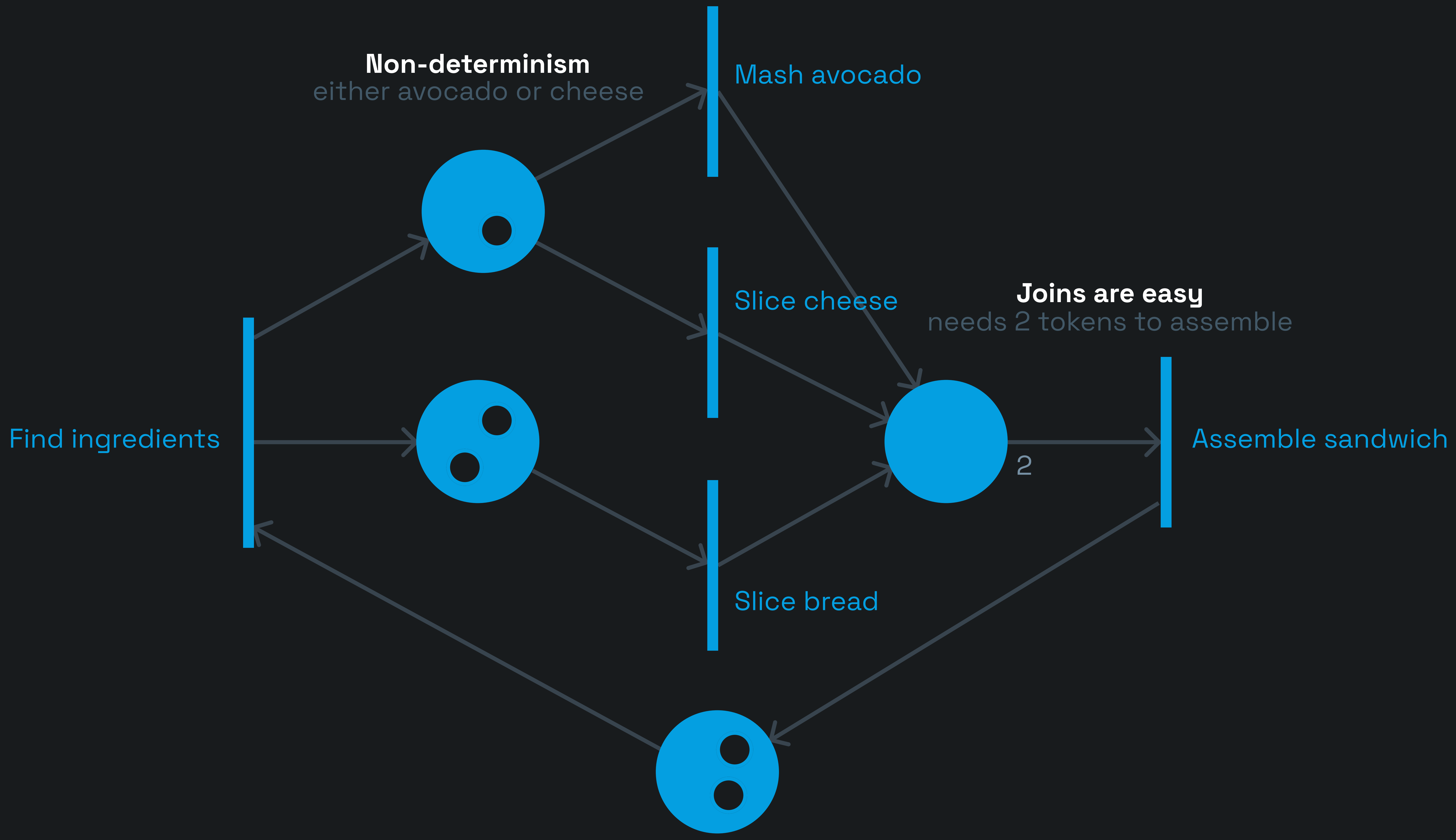
A software perspective



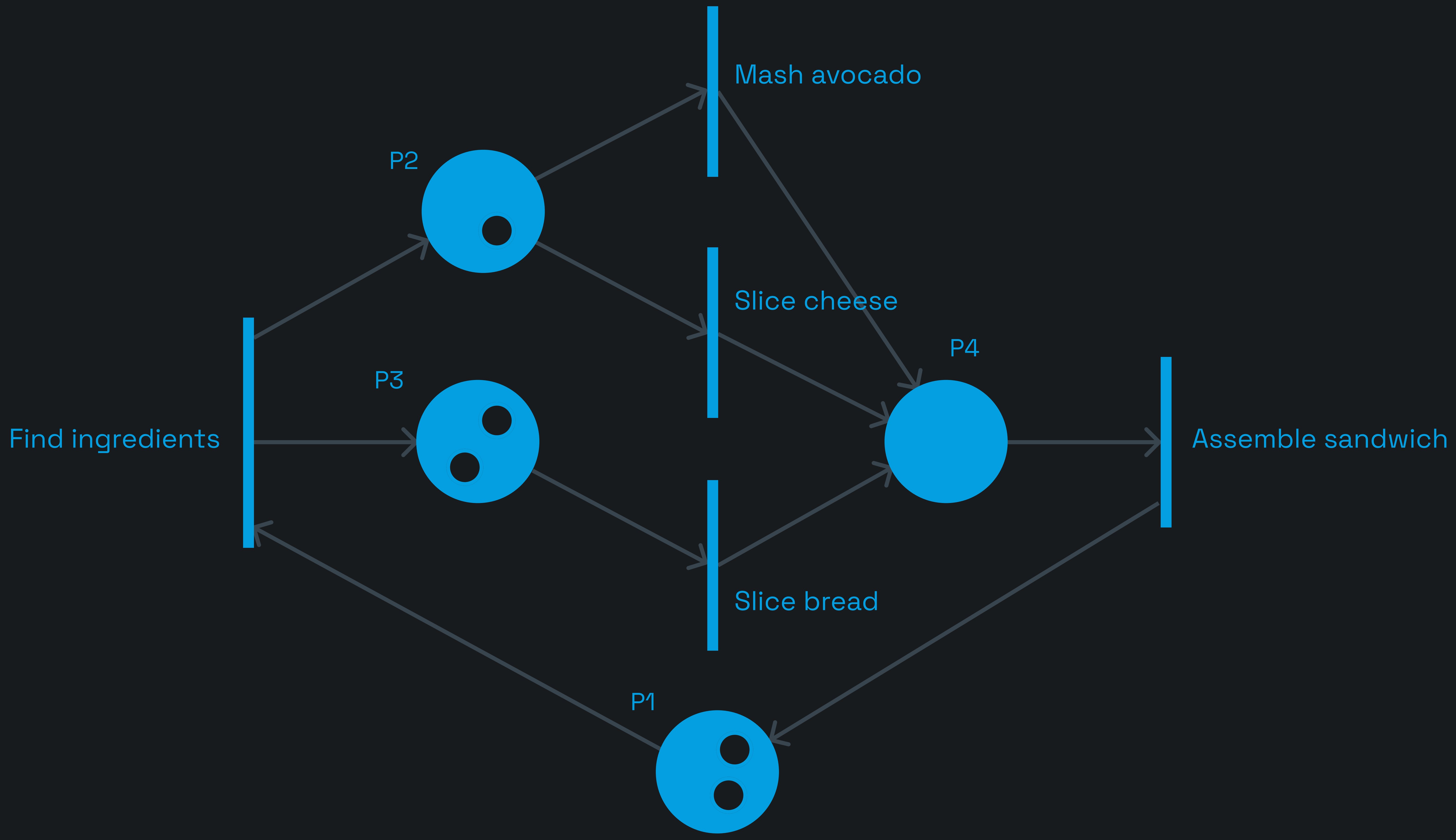
Places
enabling the functions

Transitions
the functions themselves


Mash avocado
Slice cheese
Find ingredients
Assemble sandwich
Slice bread

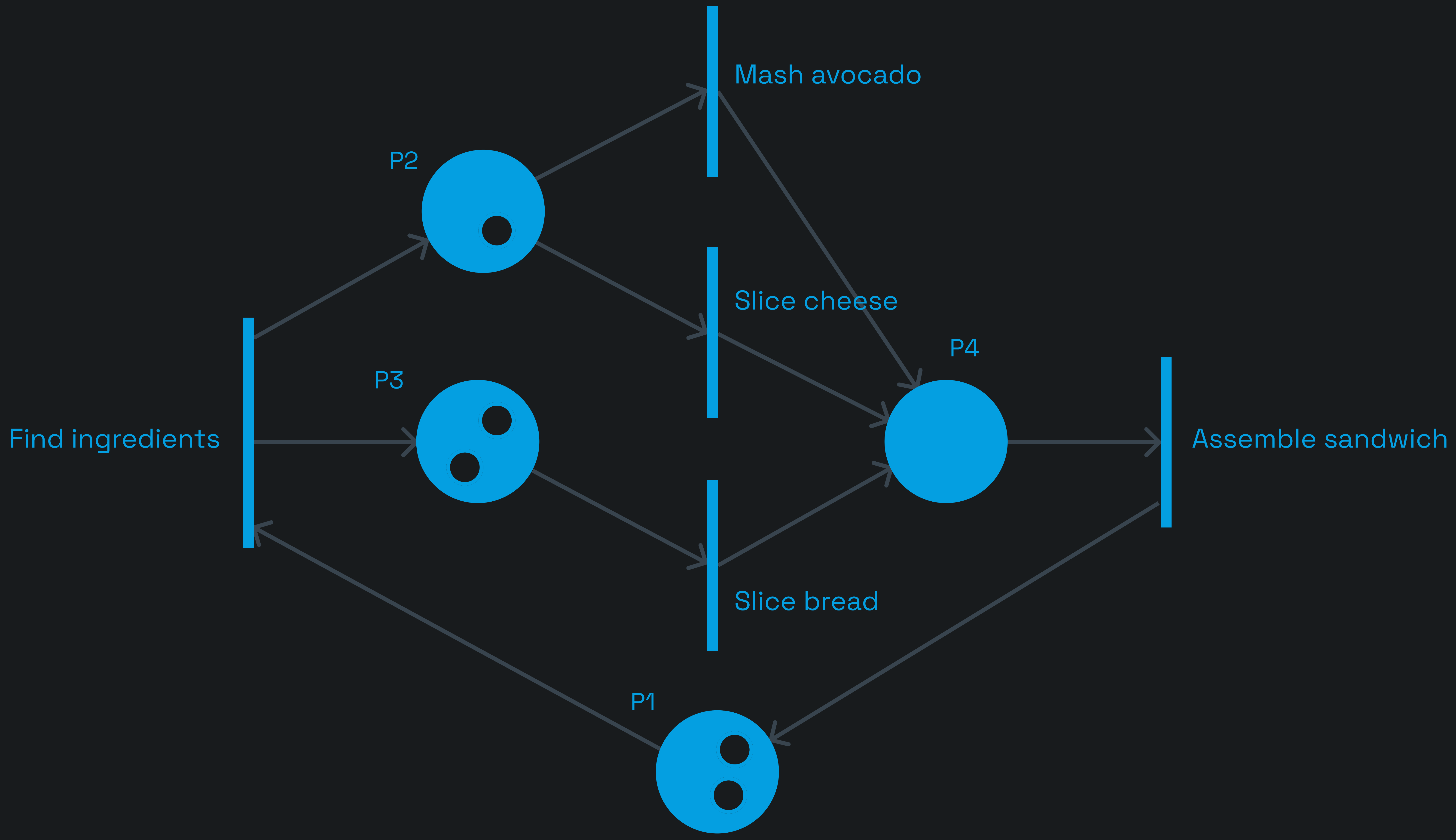

Non-determinism
either avocado or cheese
Mash avocado
Slice cheese
Joins are easy
needs 2 tokens to assemble
Find ingredients
Assemble sandwich
2
Slice bread

State (or marking)



$$M = [2, 1, 2, 0]$$

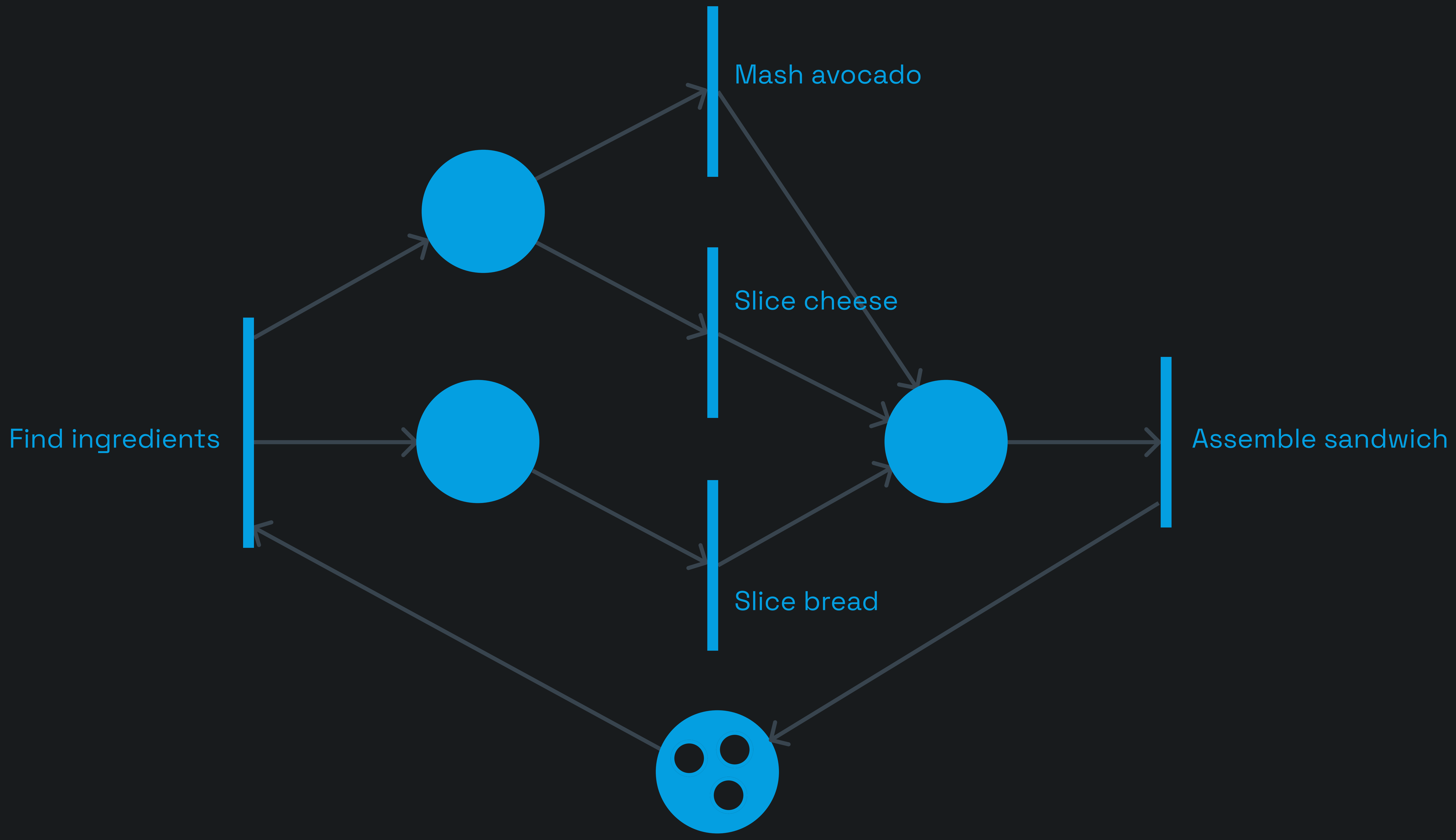
number of tokens in each place is the state of the net


Mash avocado
P2
Slice cheese
P4
P3
Find ingredients
Assemble sandwich
Slice bread
P1

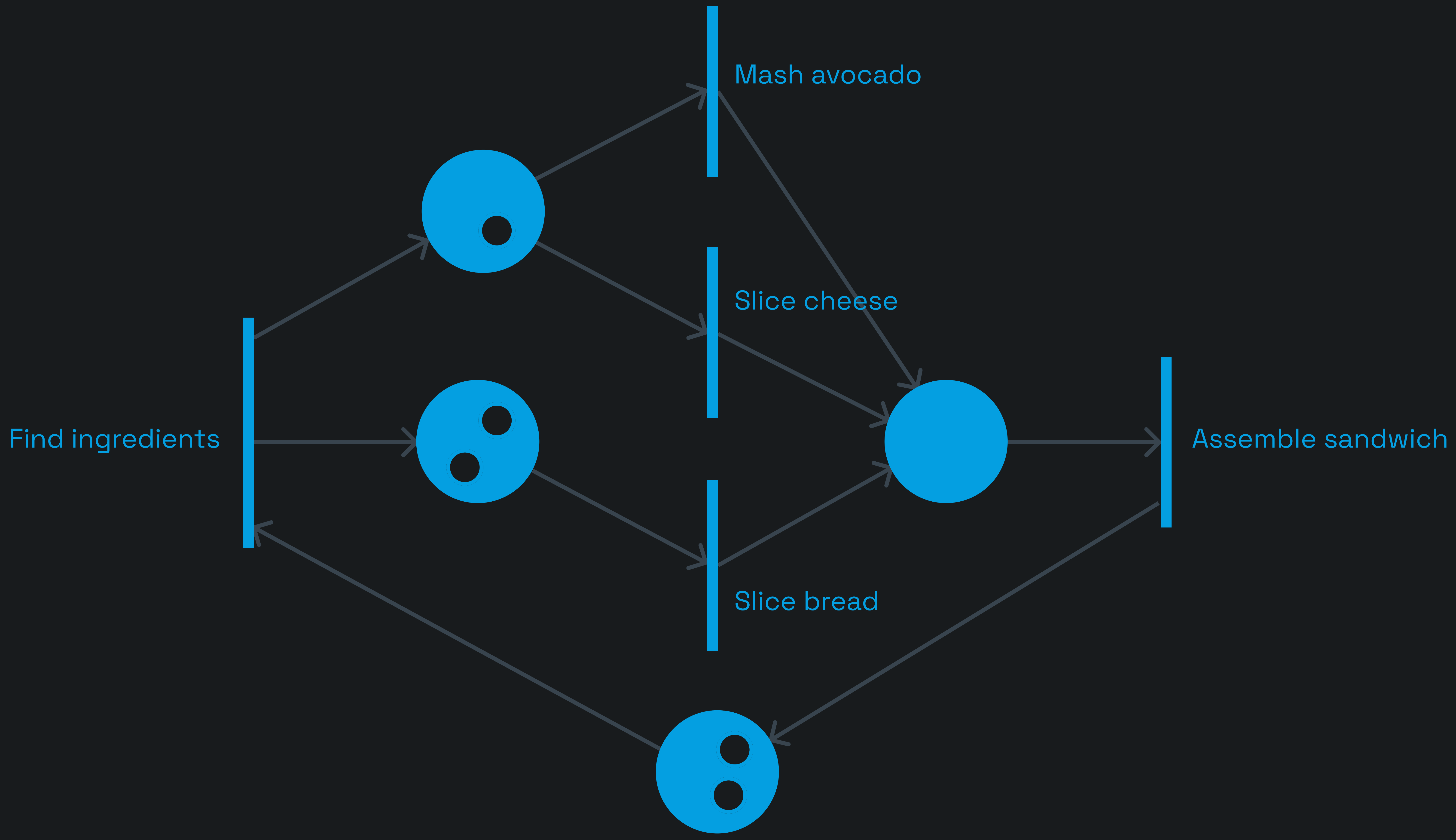
$$N = (P, T, A, W, M_0)$$

Detecting deadlocks

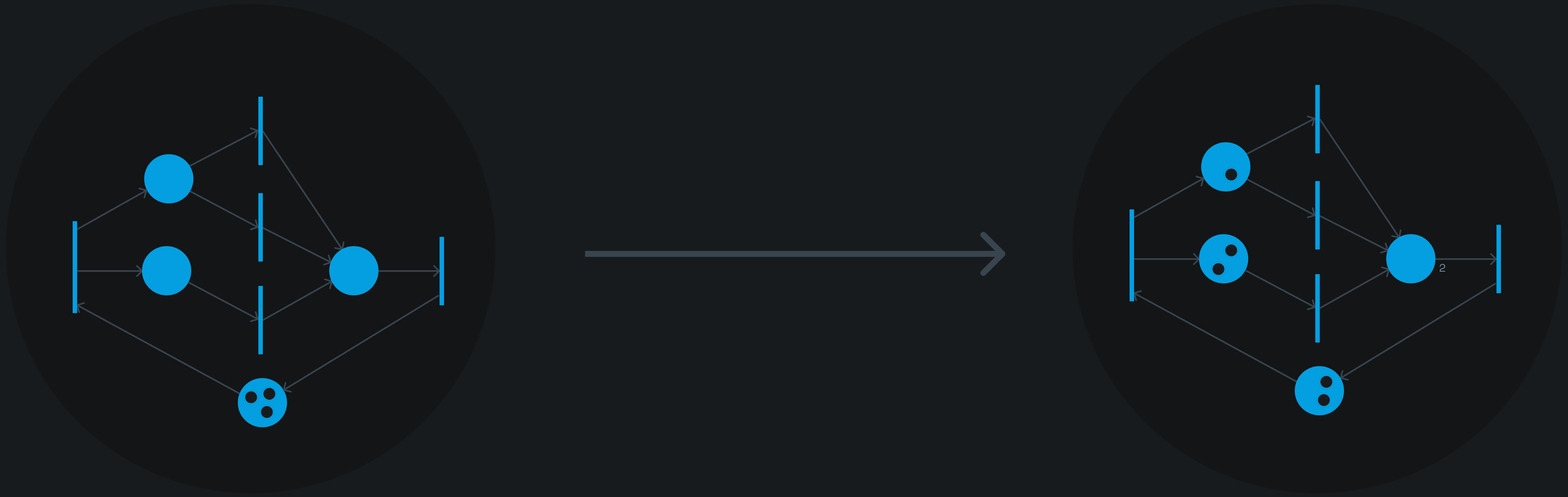
from just the initial marking M_0


Mash avocado
Slice cheese
Find ingredients
Assemble sandwich
Slice bread

$$M_0 = [3, 0, 0, 0]$$


Mash avocado
Slice cheese
Find ingredients
Assemble sandwich
Slice bread

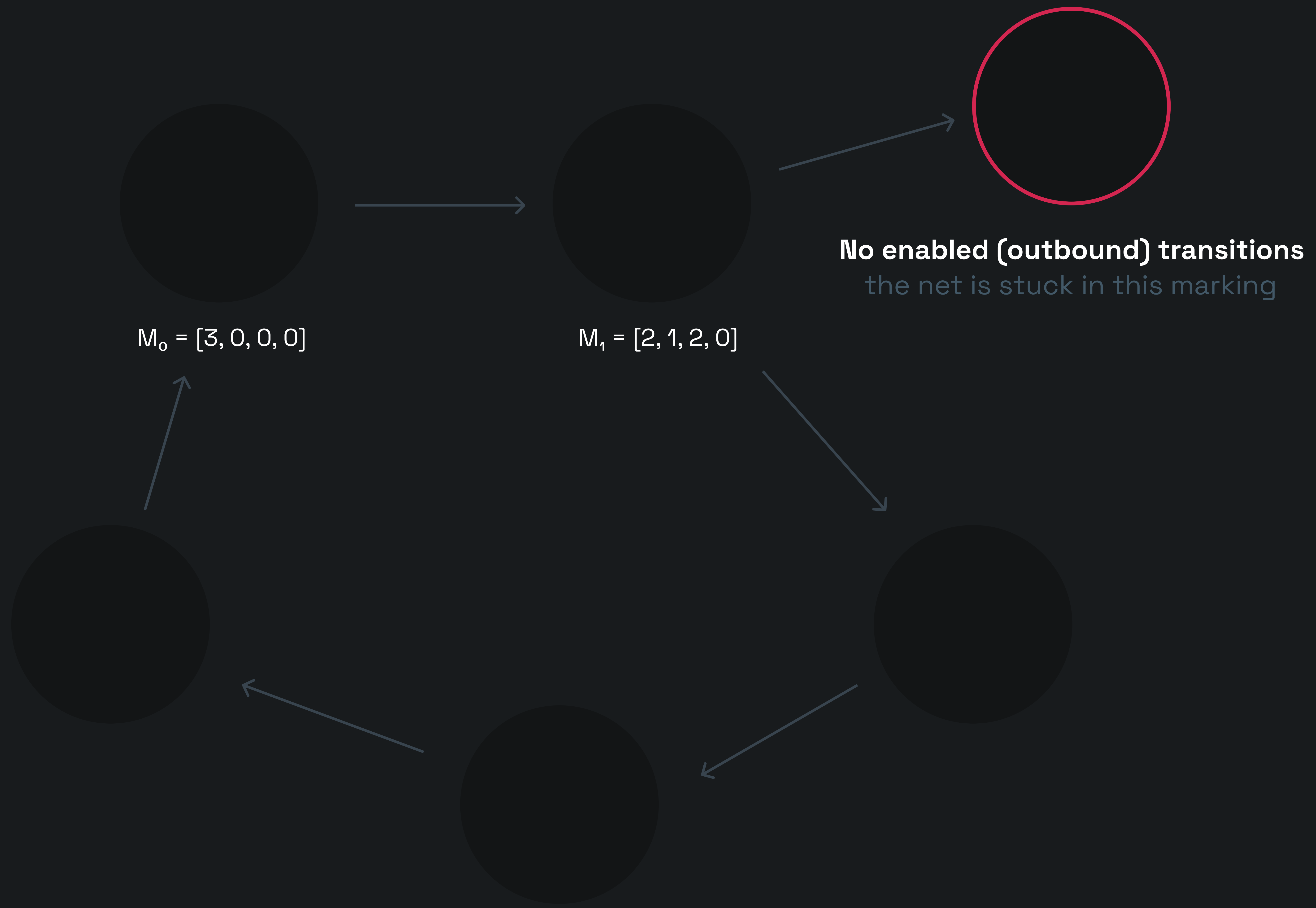
$M_1 = [2, 1, 2, 0]$



$M_0 = [3, 0, 0, 0]$

$M_1 = [2, 1, 2, 0]$


$M_0 = [3, 0, 0, 0]$
$M_1 = [2, 1, 2, 0]$



No enabled (outbound) transitions
the net is stuck in this marking
M_0 = [3, 0, 0, 0]
M_1 = [2, 1, 2, 0]

compile-time deadlock detection ..

too good to be true?

Petri Net reachability analysis

EXPSPACE

due to the **state explosion** problem

RESEARCH-ARTICLE

The Reachability Problem for Petri Nets Is Not Elementary

Authors: Wojciech Czerwiński, Sławomir Lasota, Ranko Lazić, JÉrôme Leroux, and Filip Mazowiecki | Authors Info & Claims

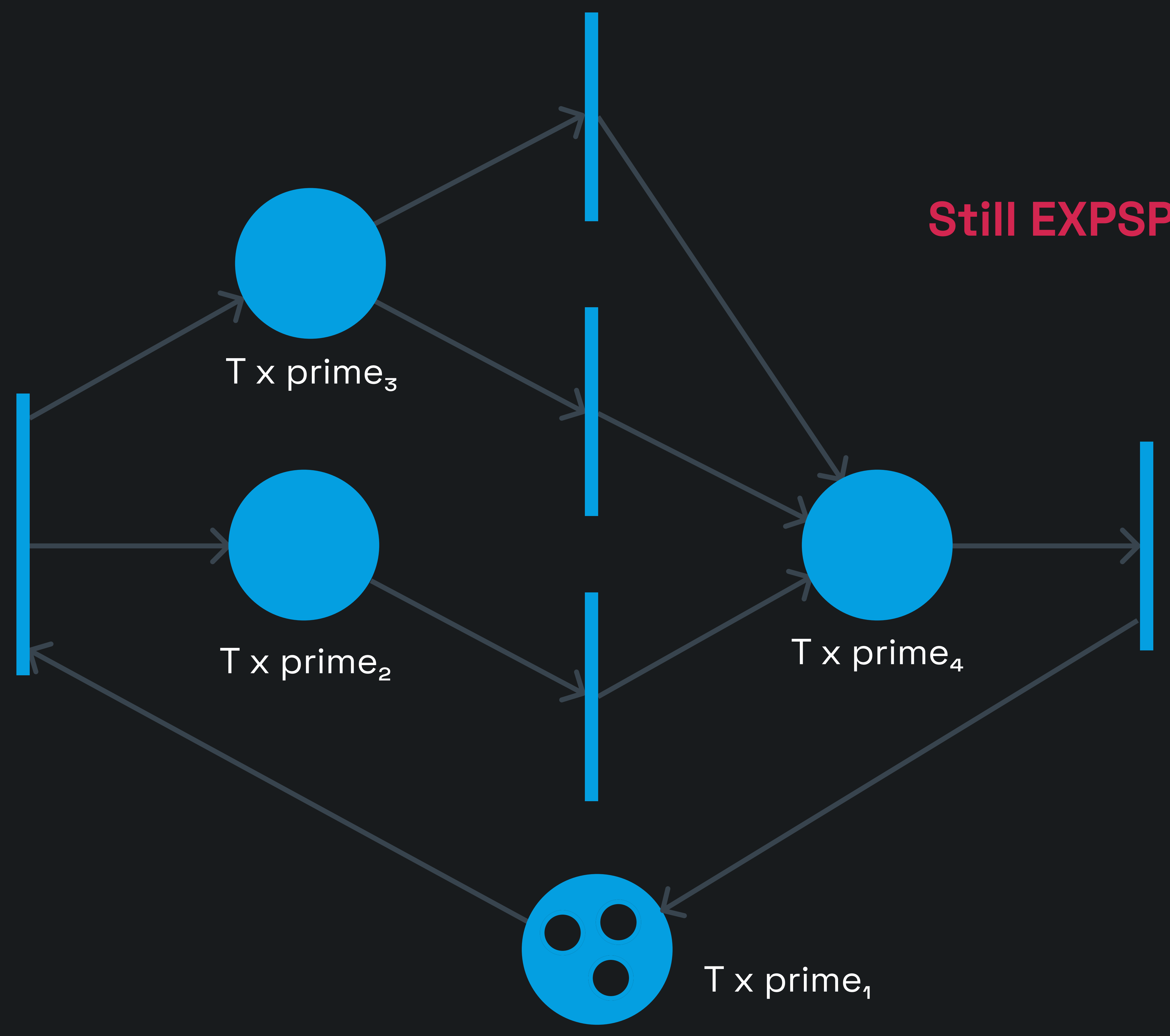
Journal of the ACM (JACM), Volume 68, Issue 1 • Article No.: 7, Pages 1 - 28 • https://doi.org/10.1145/3422822

Published: 22 December 2020 Publication History

of the problem has remained unsettled since the 1960s, and it is one of the most prominent open questions in the theory of verification. Decidability was proved by Mayr in his seminal STOC 1981 work, and, currently, the best published upper bound is non-primitive recursive Ackermannian of Leroux and Schmitz from Symposium on Logic in Computer Science 2019. We establish a non-elementary lower bound, i.e., that the reachability problem needs a tower of exponentials of time and space. Until this work, the best lower bound has been exponential space, due to Lipton in 1976. The new lower bound is a major breakthrough for several reasons. Firstly, it shows that the reachability problem is much harder than the coverability (i.e., state reachability) problem, which is also ubiquitous but has been known to be complete for exponential

Partial hashing using the FTA

hash and un-hash based on unique primes and places that changed



Still EXPSPACE

Z-Net

Modifying petri nets with shady math for polynomial-time deadlock detection

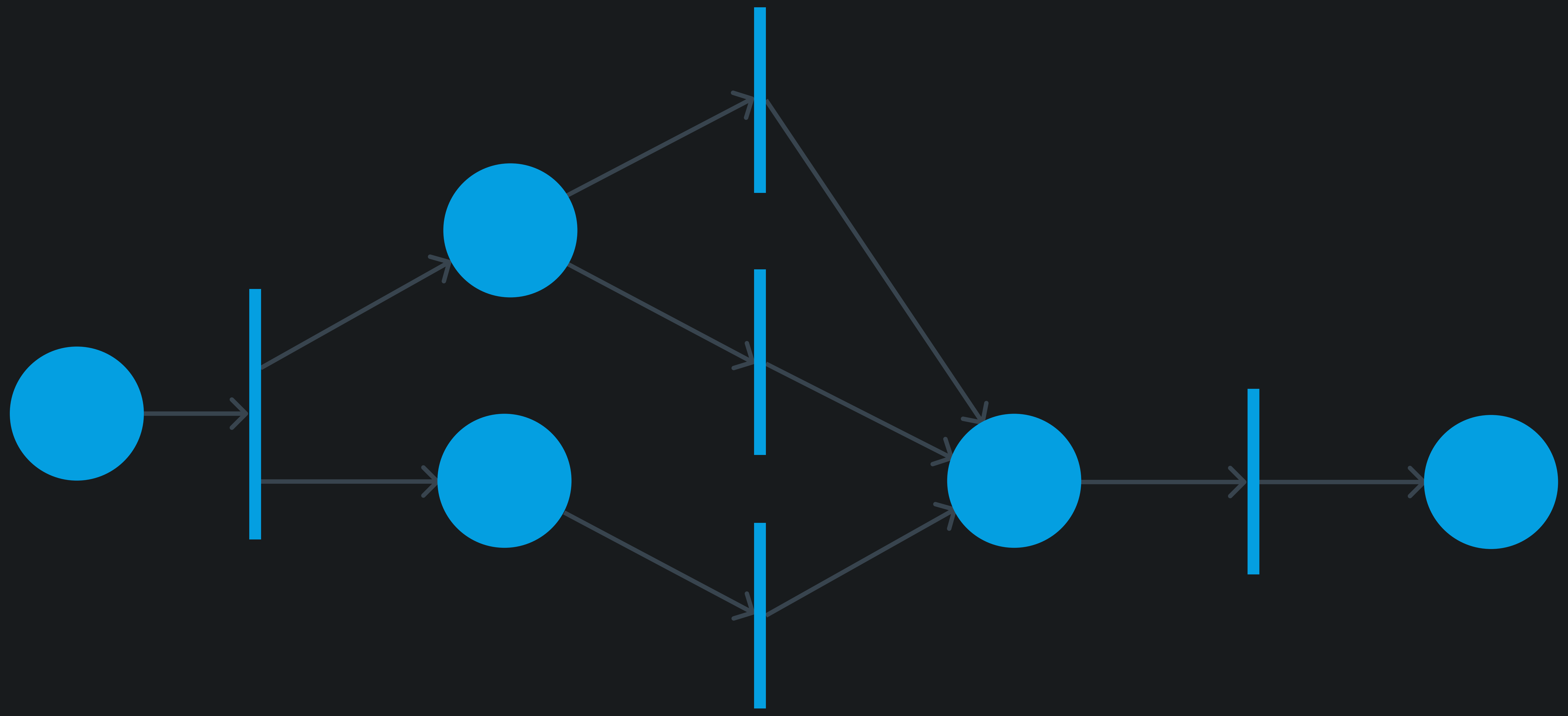
Goals

- Polynomial time deadlock detection
- **Recursion*** and free-choice

*causes state explosion by default

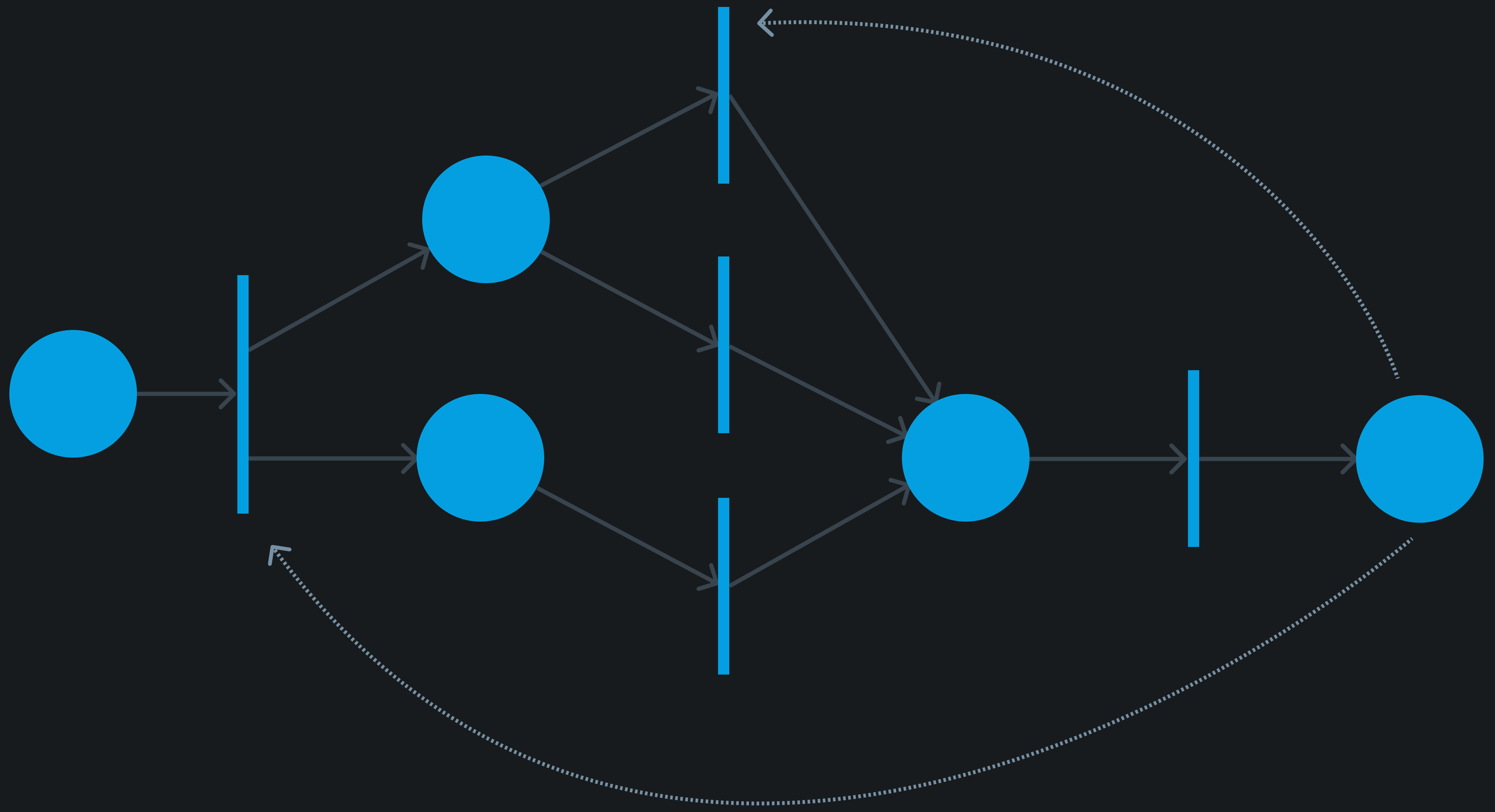
Reachability in acyclic petri nets is manageable

but how do we model recursion?

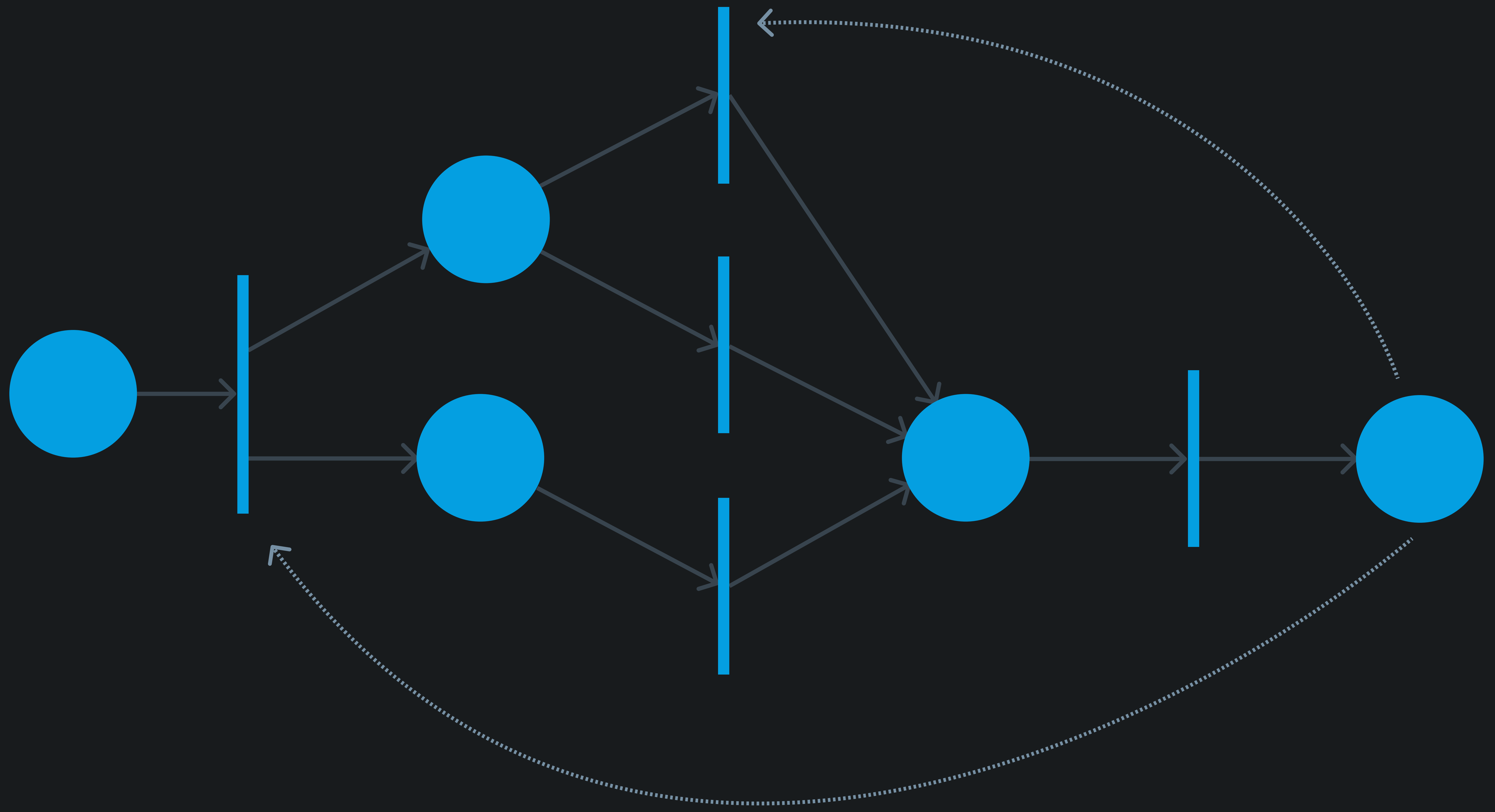


Reachability in acyclic petri nets is manageable

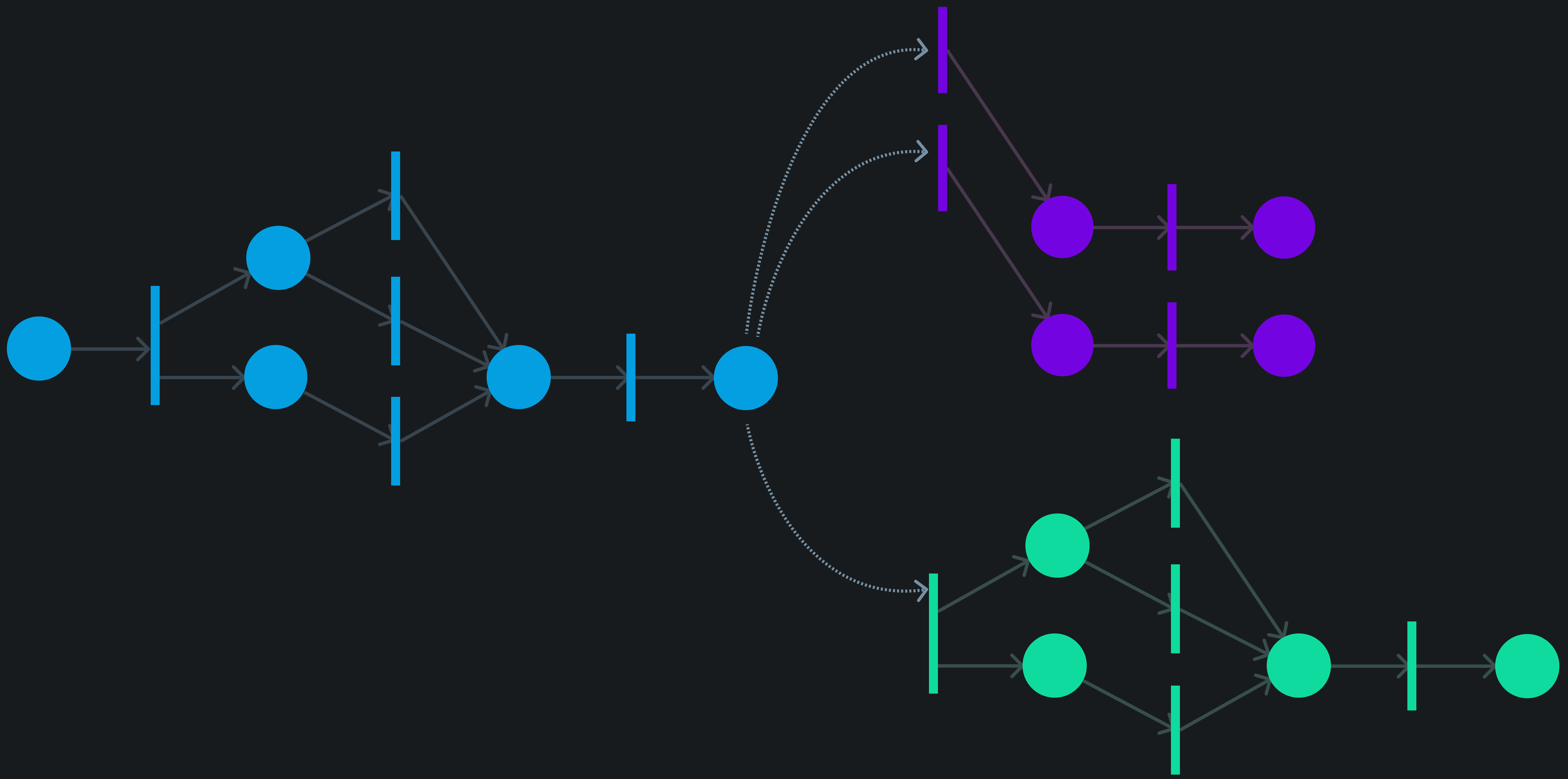
but how do we model recursion?



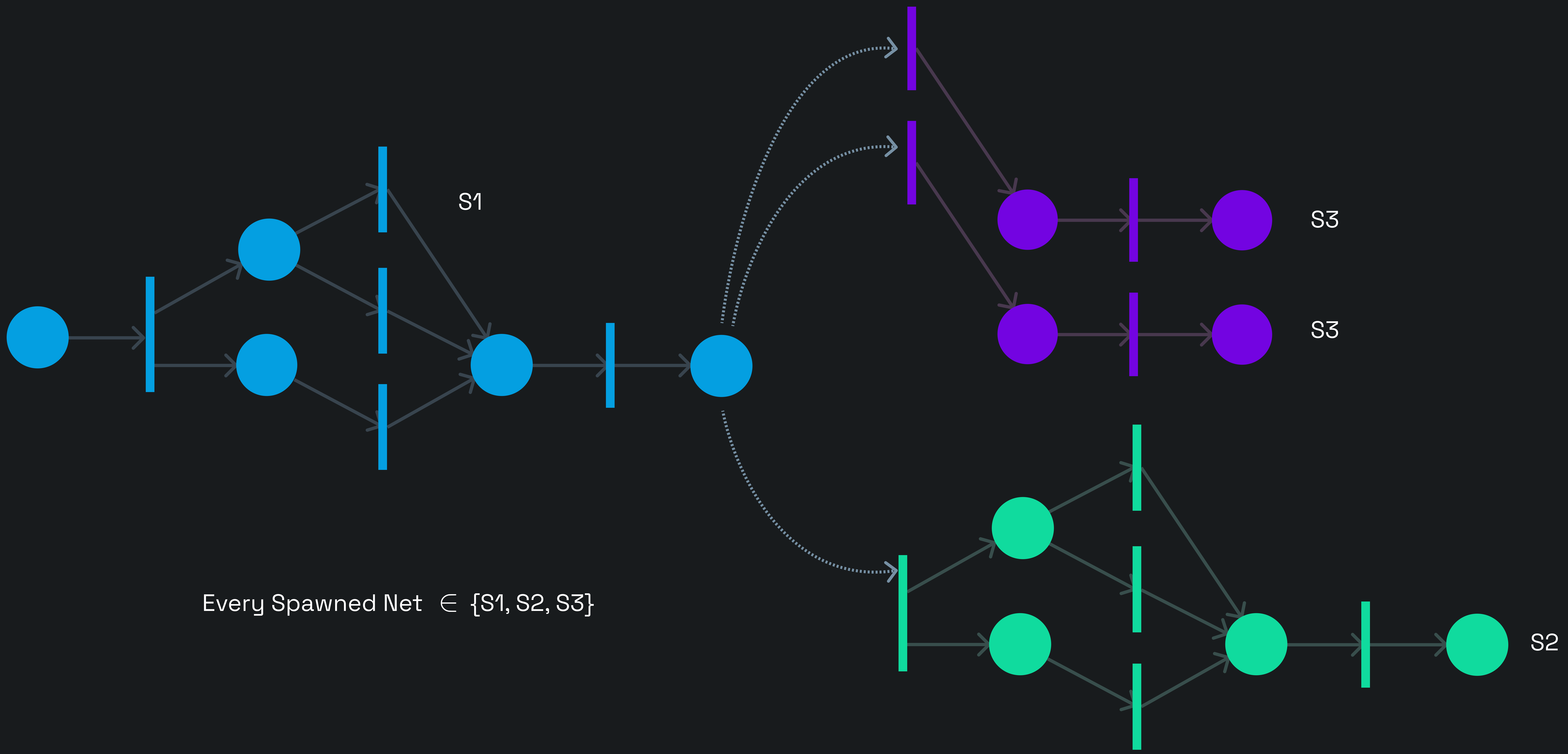
... by not using recursion, and spawning **state-independent** subnets

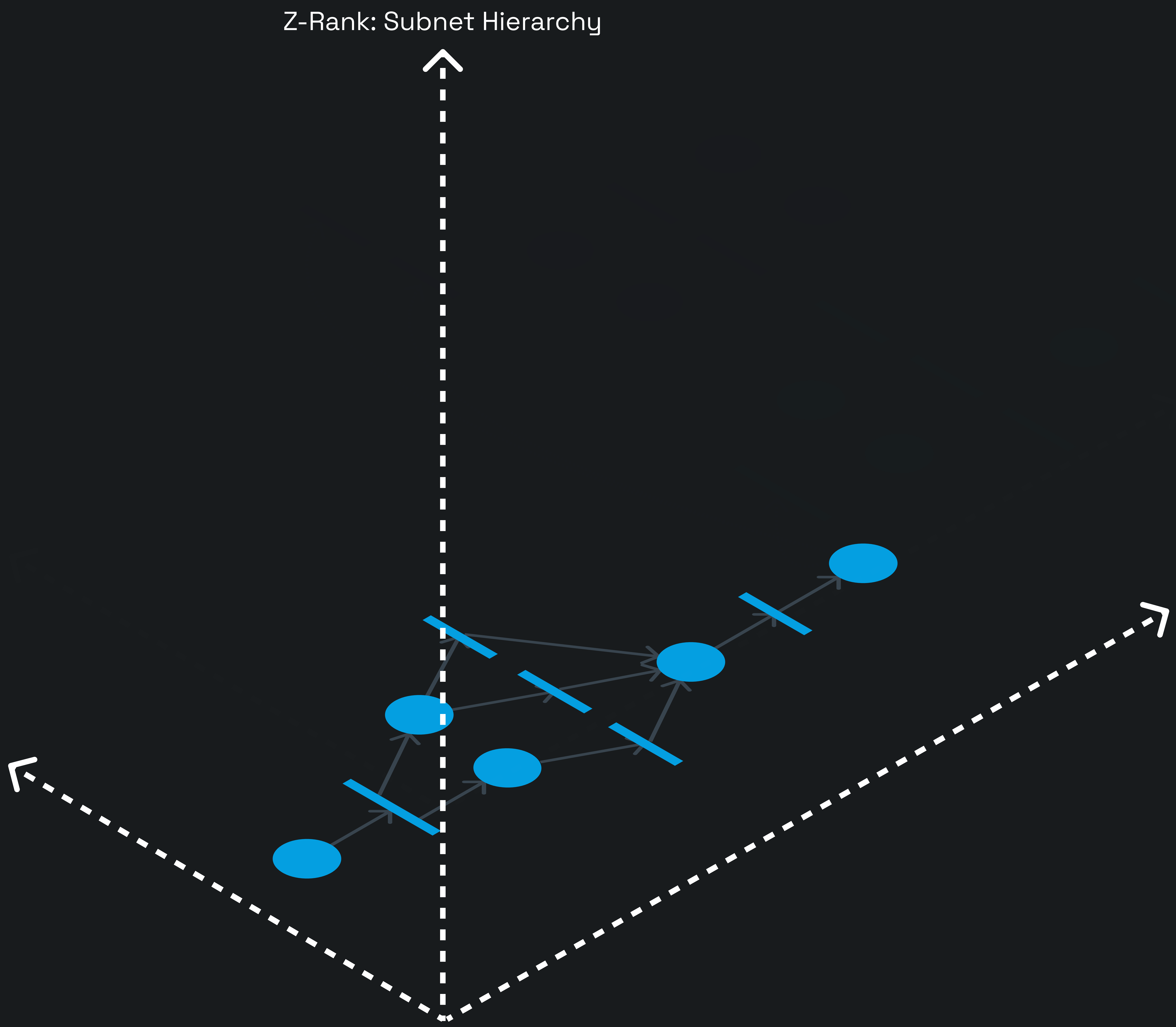


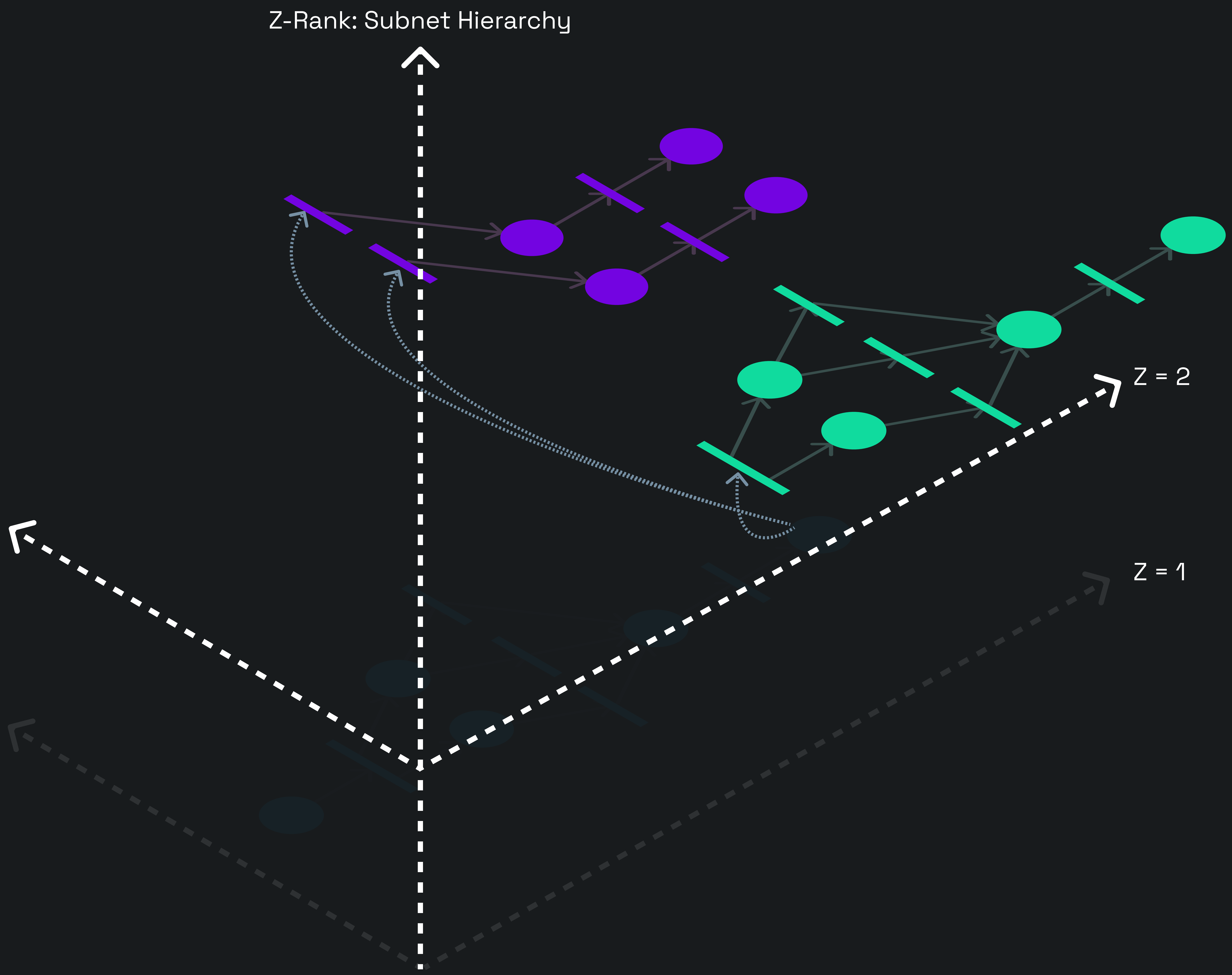
... by not using recursion, and spawning **state-independent** subnets



each spawned net (there can be infinite), is from a finite set of unique subnets

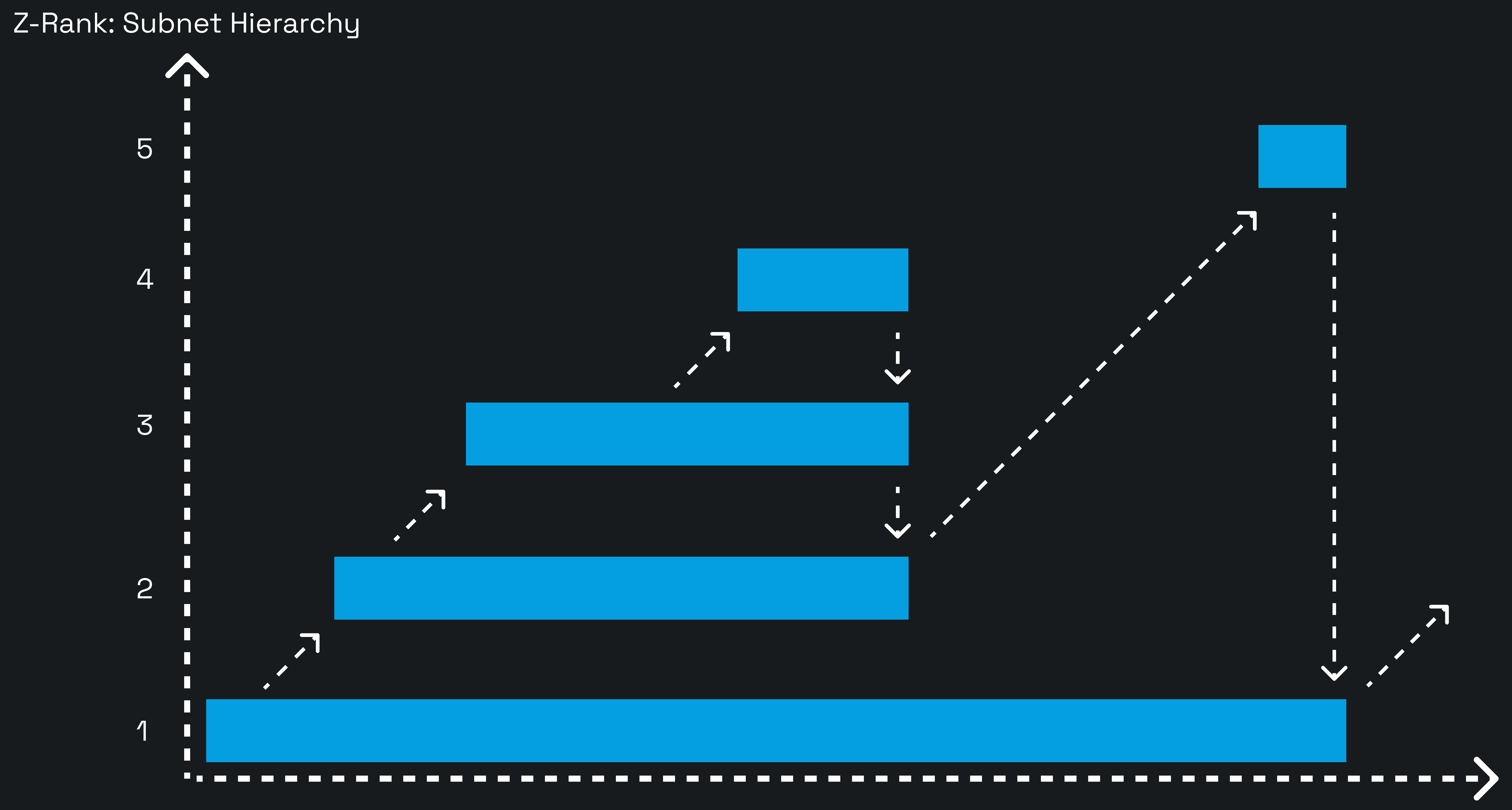



Z-Rank: Subnet Hierarchy


Z-Rank: Subnet Hierarchy
Z = 2
Z = 1

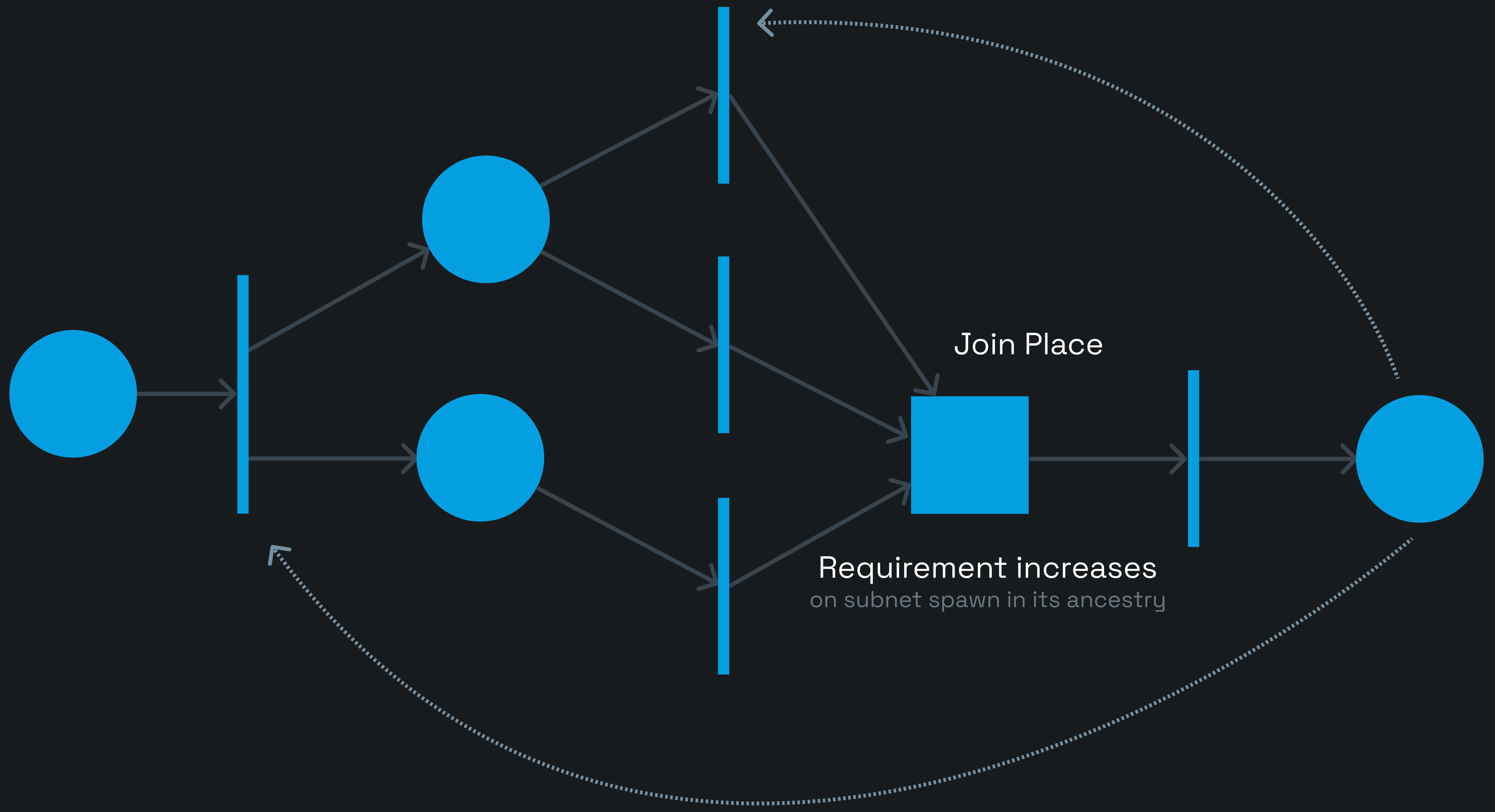
Cross-dimensional synchronization

waiting for spawned subnets to finish



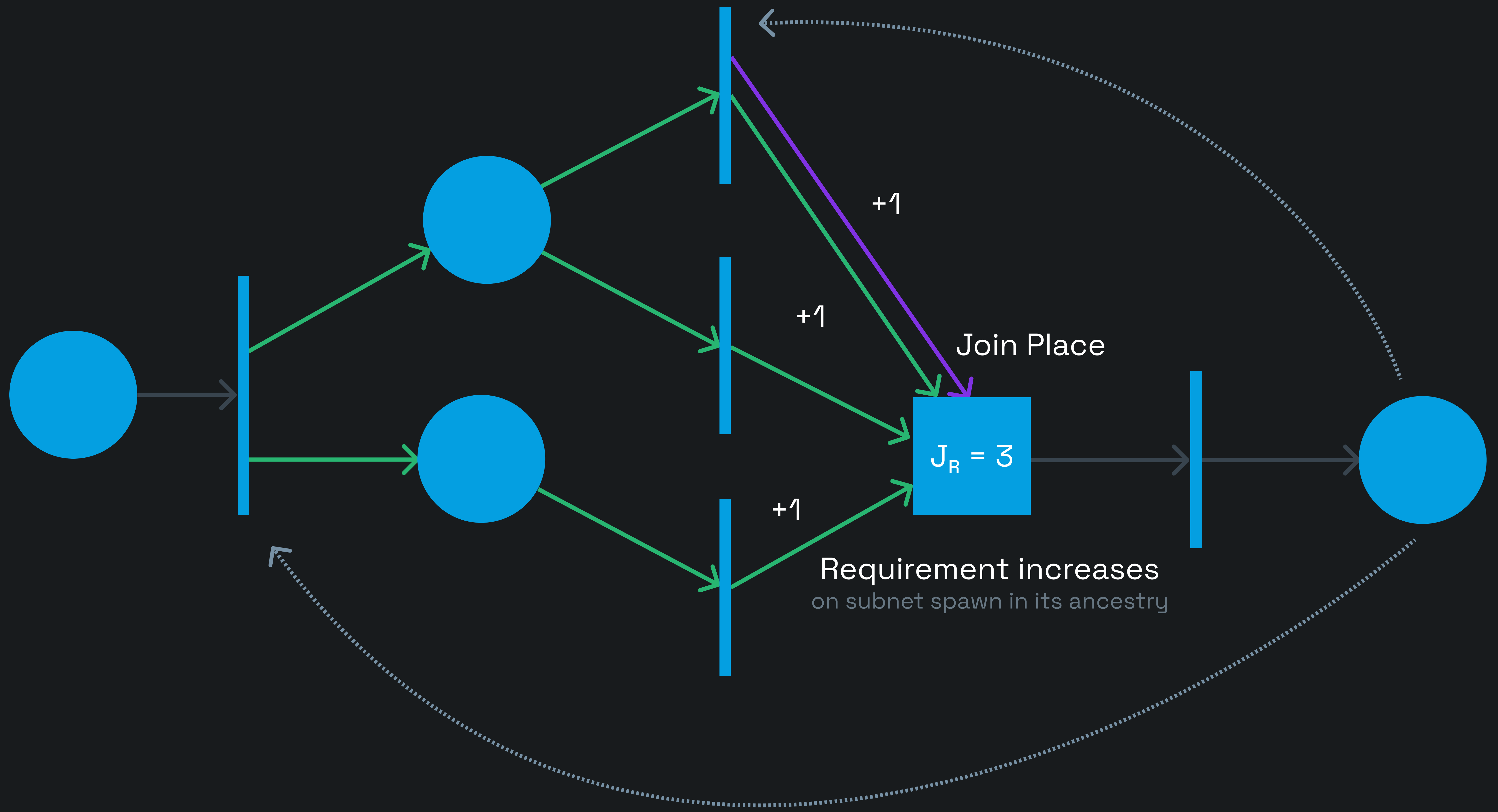
Cross-dimensional synchronization

waiting for spawned subnets to finish



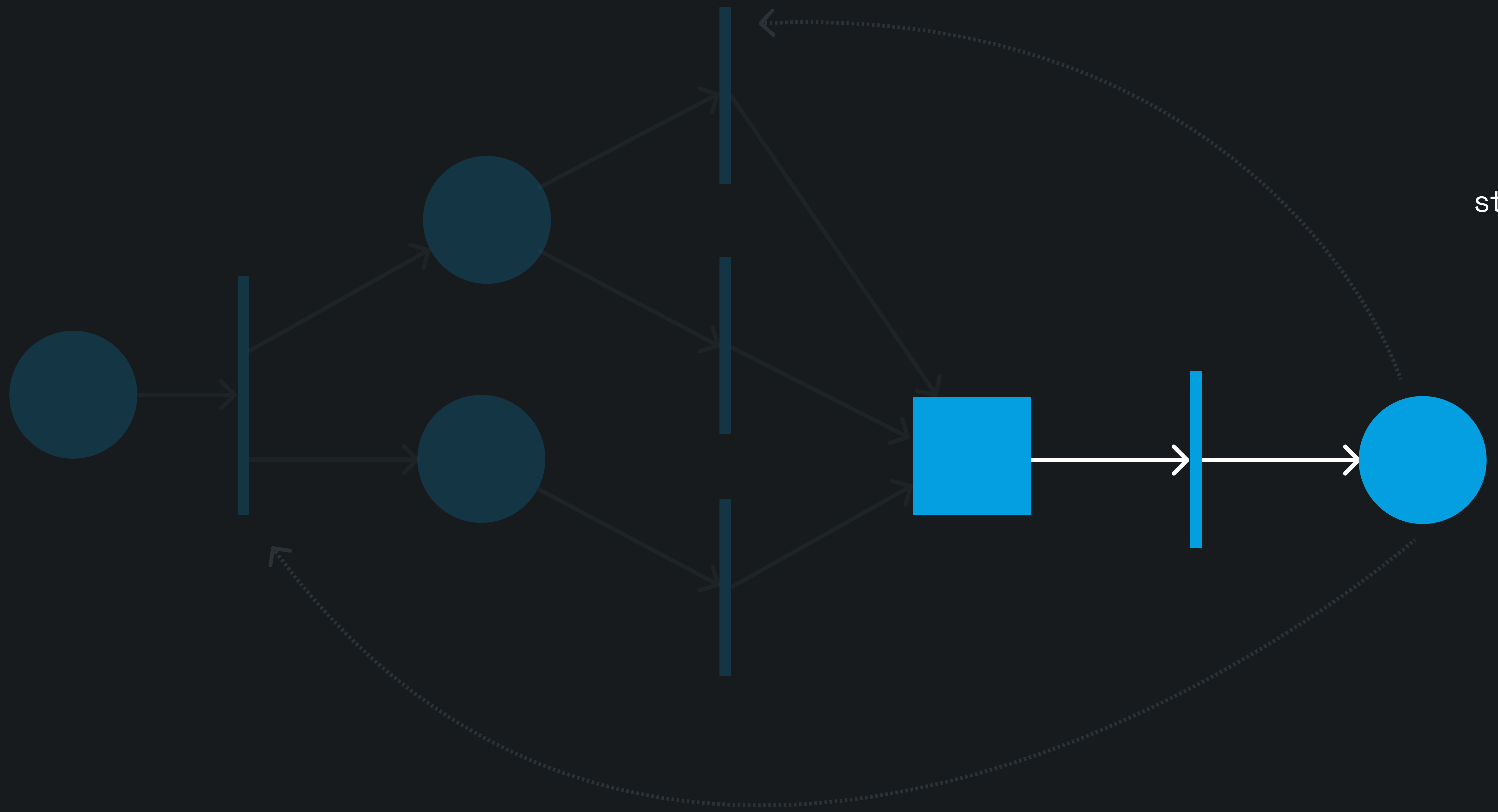
Cross-dimensional synchronization

waiting for spawned subnets to finish



Cross-dimensional synchronization

waiting for spawned subnets to finish

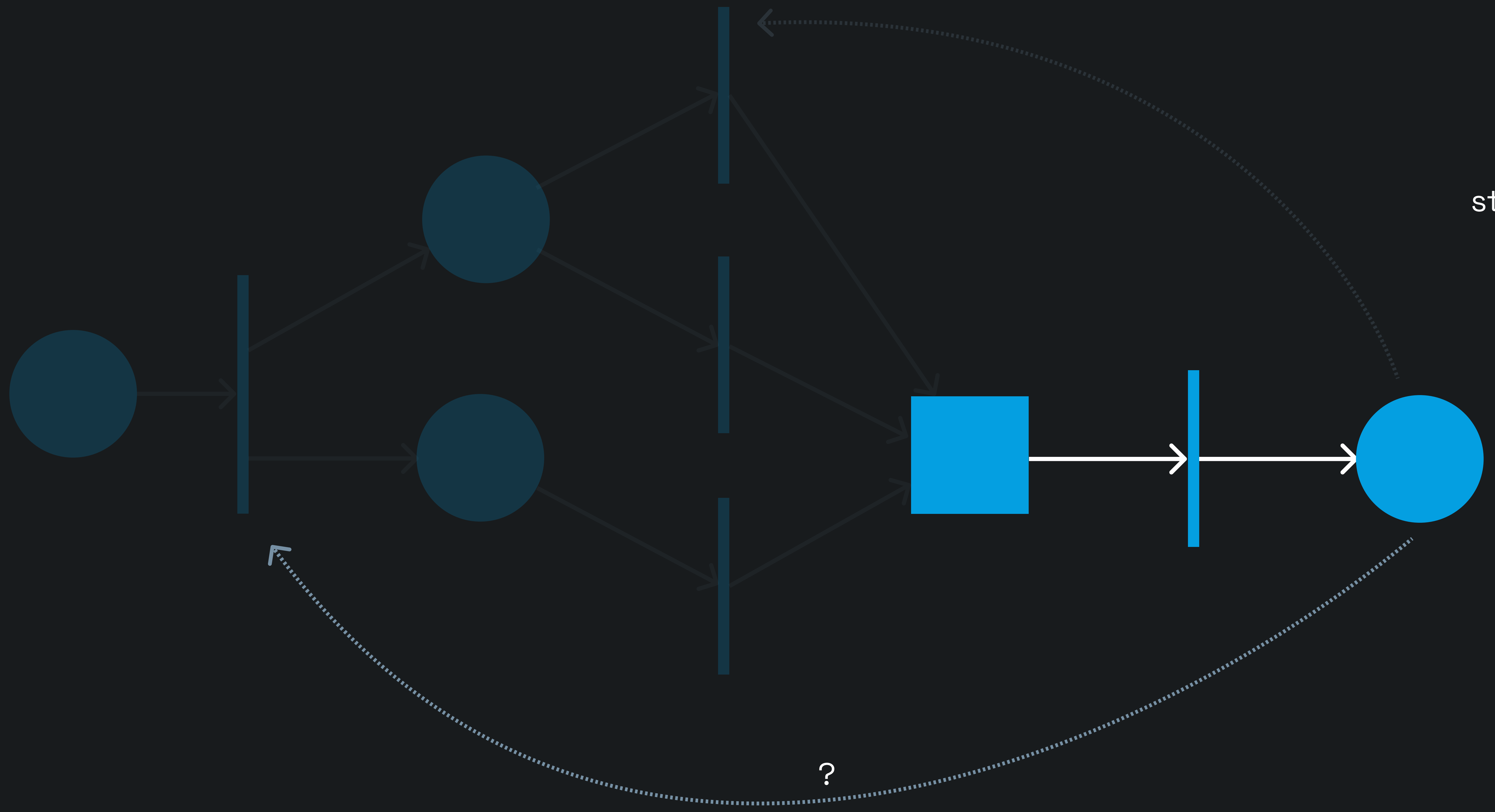


Once joined, spawn new
state-refreshed subnet at base Z

it itself is a root node too

Cross-dimensional synchronization

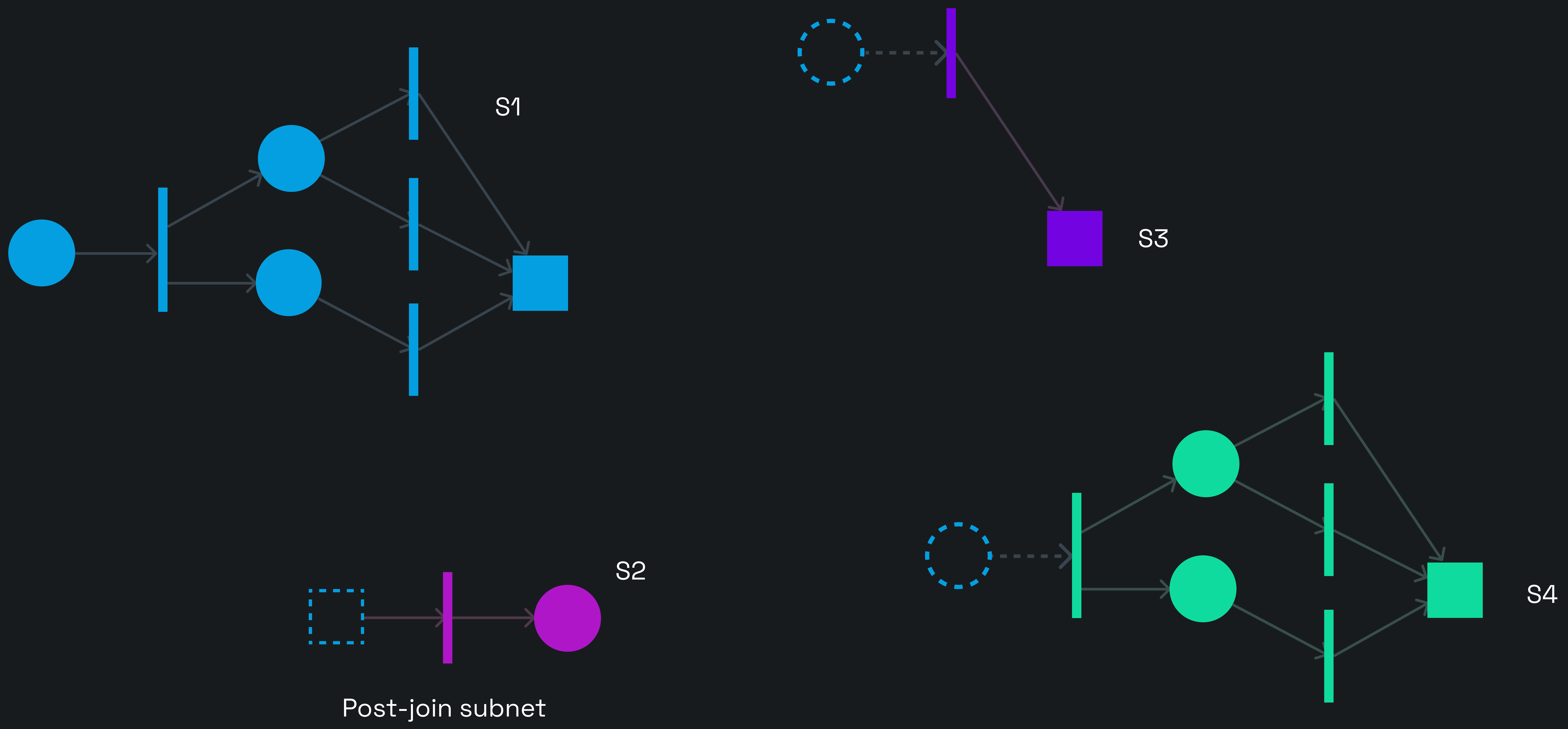
waiting for spawned subnets to finish



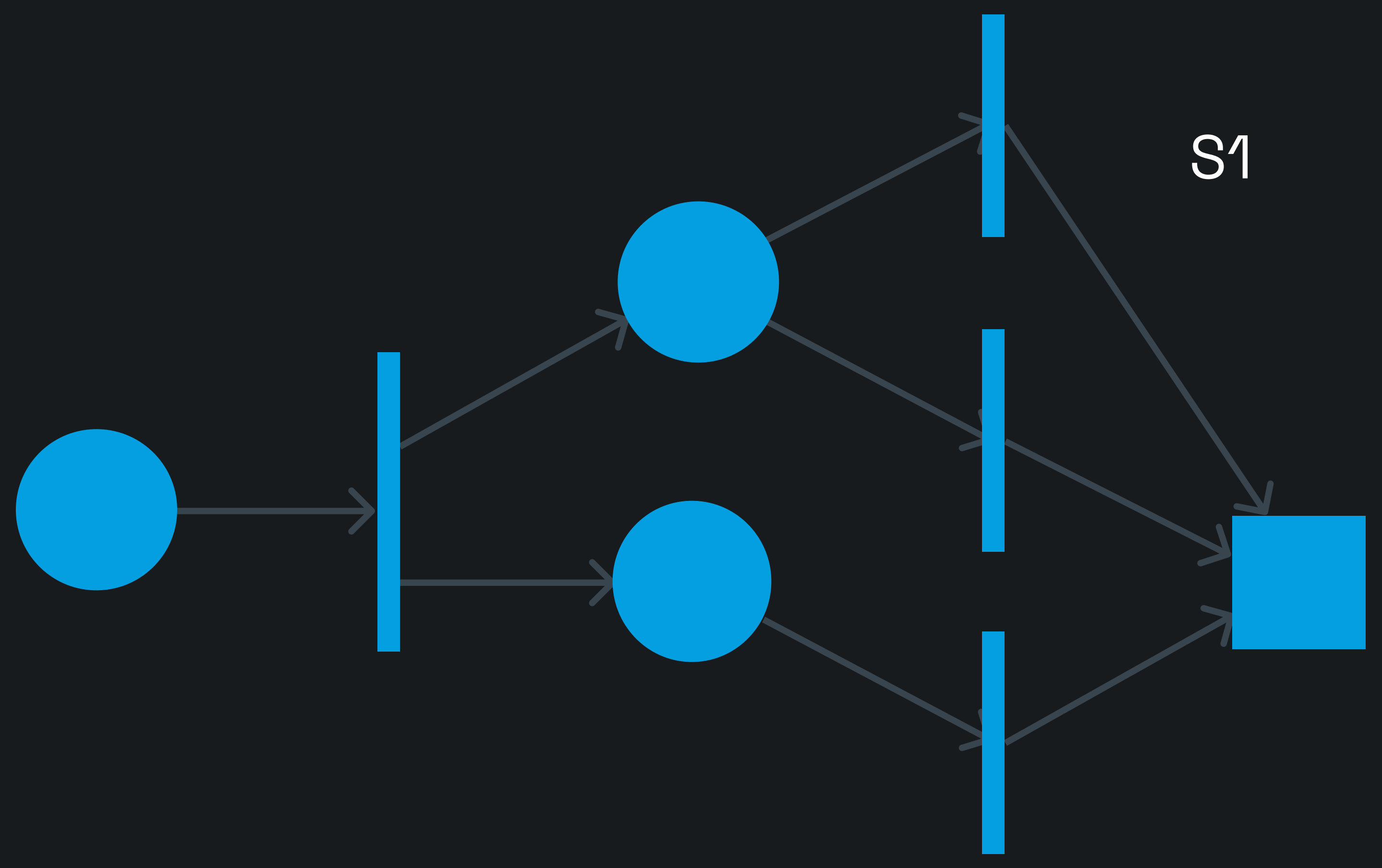
Once joined, spawn new
state-refreshed subnet at base Z
it itself is a root node too

As it can fire multiple times

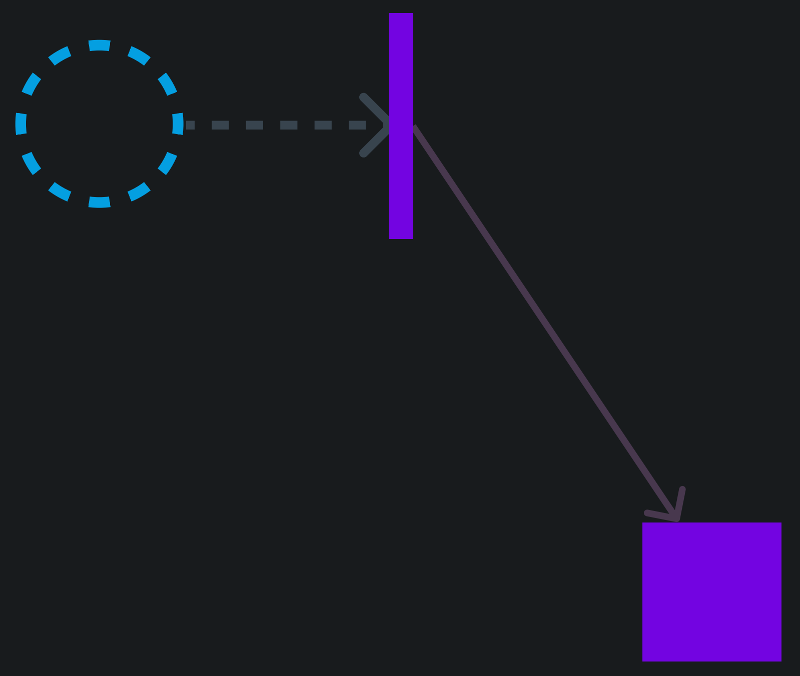
reachability for every unique acyclic subnet independently in polynomial time



reachability for every unique acyclic subnet independently in polynomial time

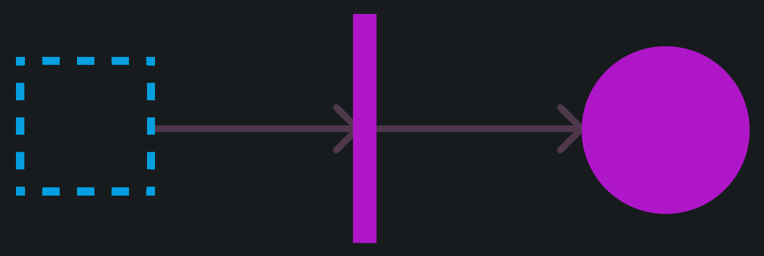


S1



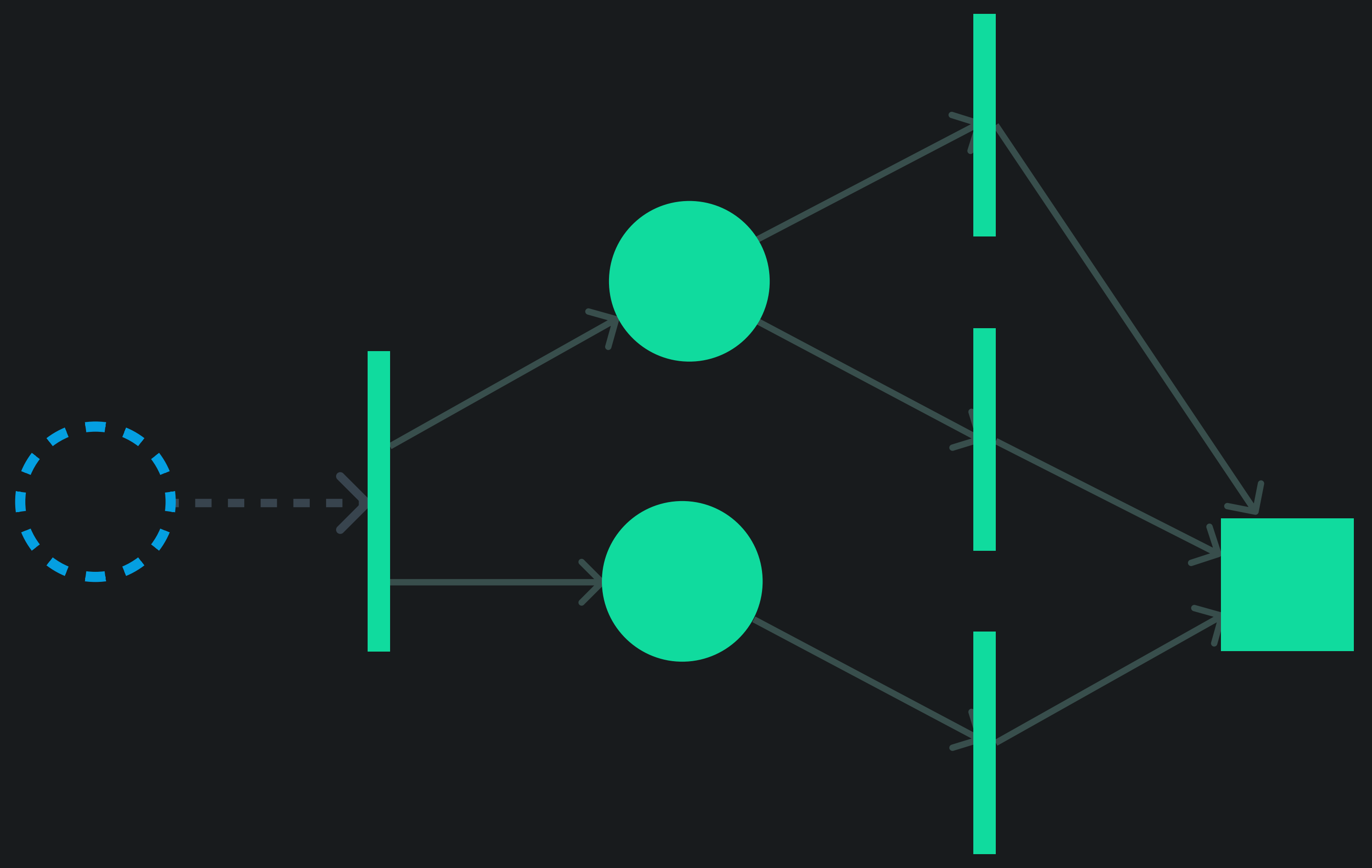
S3

$$N \in \{S1, S2, S3, S4\}$$



S2

Post-join subnet



S4

Z-Net

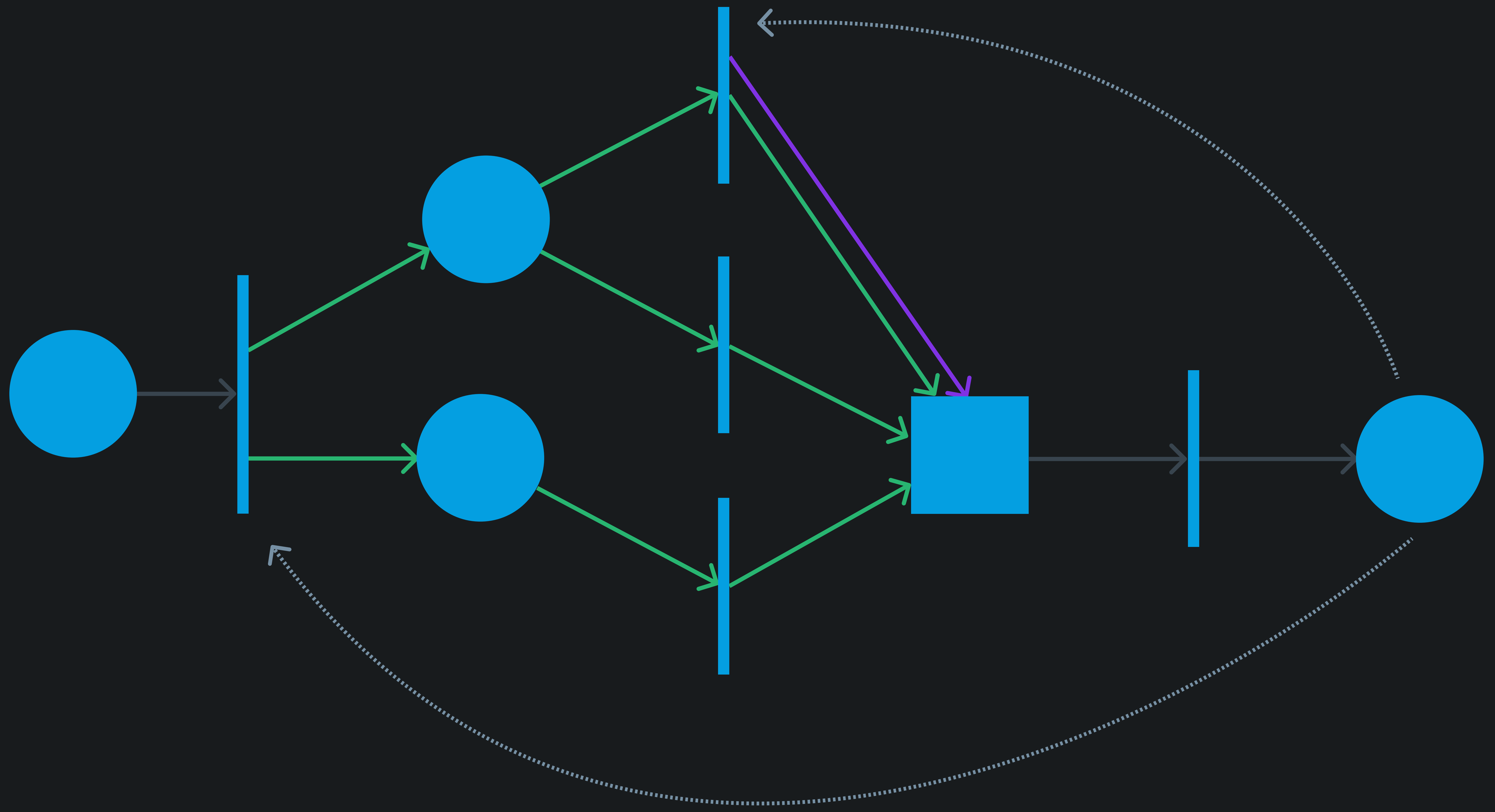
$$N_Z = (P, J, T, A, B, W, M_0)$$

- Vertices
- Edges

compile-time magic

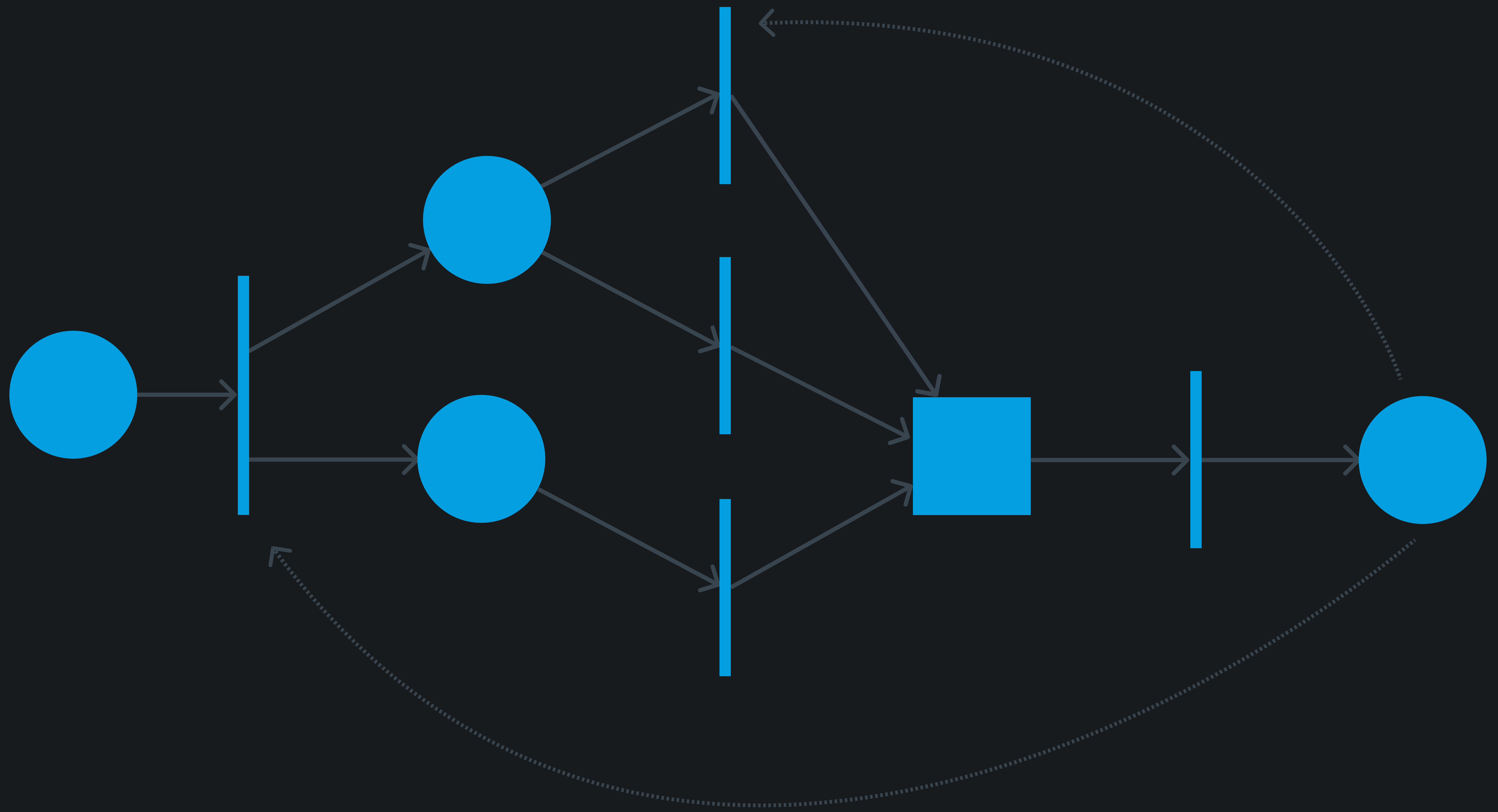
Step 1: join path resolution

traversing backwards from join node and marking each arc



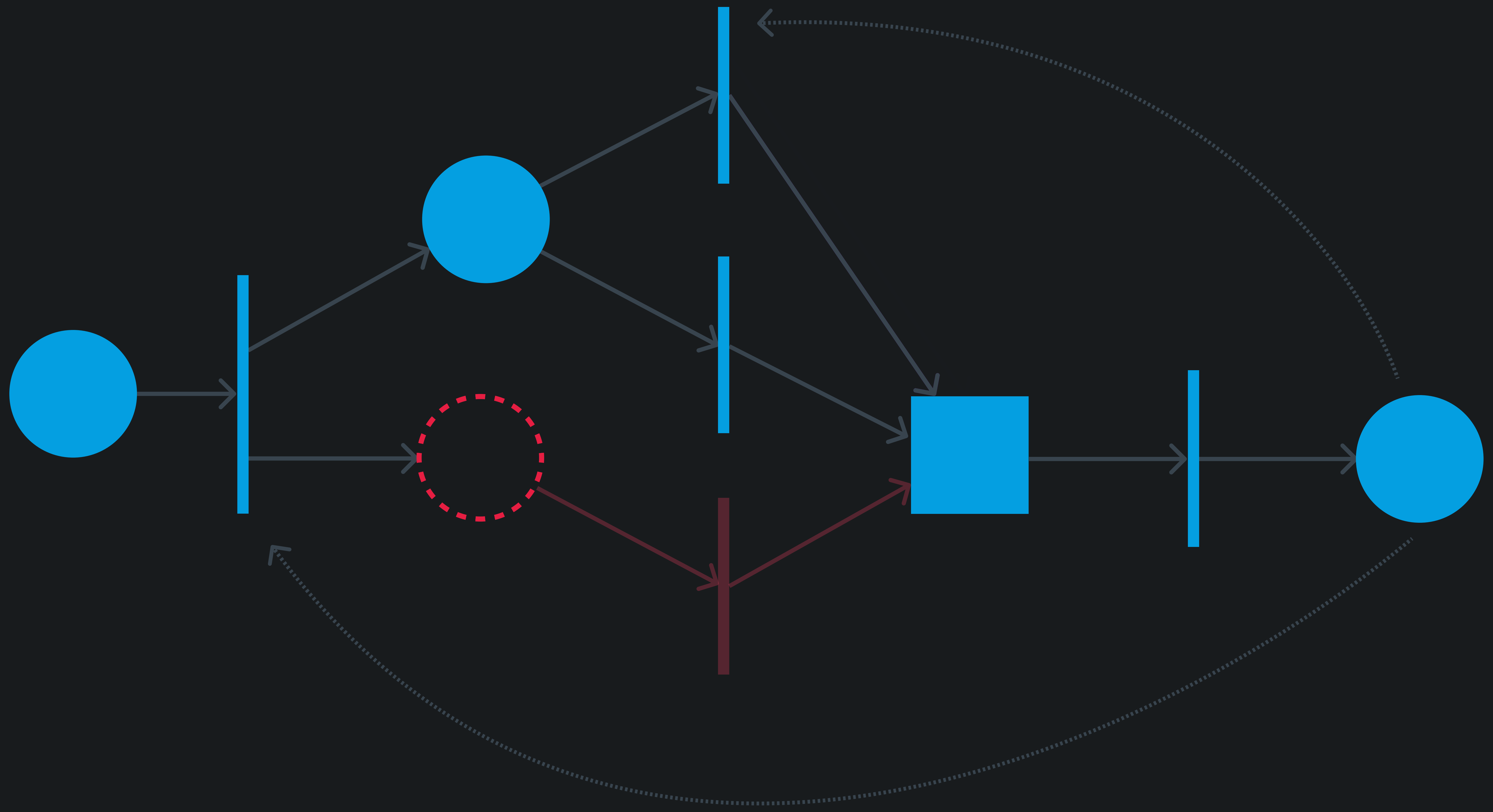
Step 2: subnet deadlock check

sort topologically, then verify all transitions are enabled by input places + check backward arcs



Extra step? Centrality measures for fault tolerance

benefits of acyclic petri nets can go beyond reachability



but a bit beyond this presentation's scope

A bit of C++

```cpp
using System = Net<
  Places<...>,
  JoinPlaces<...>,
  Transitions<...>,
  Arcs<...>,
  BackwardArcs<...>
>;

static_assert(System::is_deadlock_free);


System system;
system.bind(T0{}, foo);
system.bind(T1{}, bar);

system.start();
```

The end

https://github.com/bdasgupta02/znet
(the runtime part is in progress)