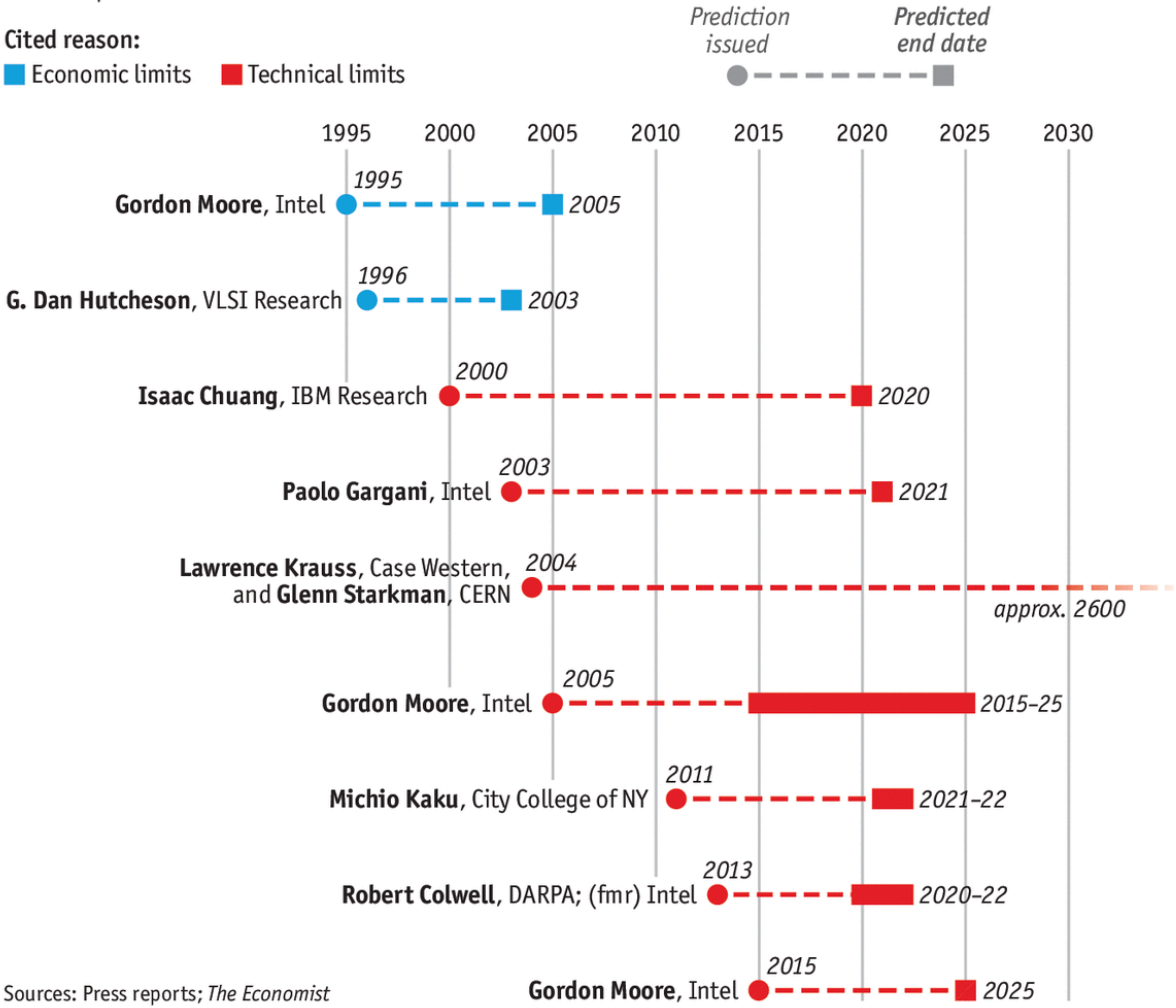
A Synthesis-Aided Compiler for Manycore Computation

emergent trend in computer architecture:

emergent trend in computer architecture:

end of Moore’s law?!?

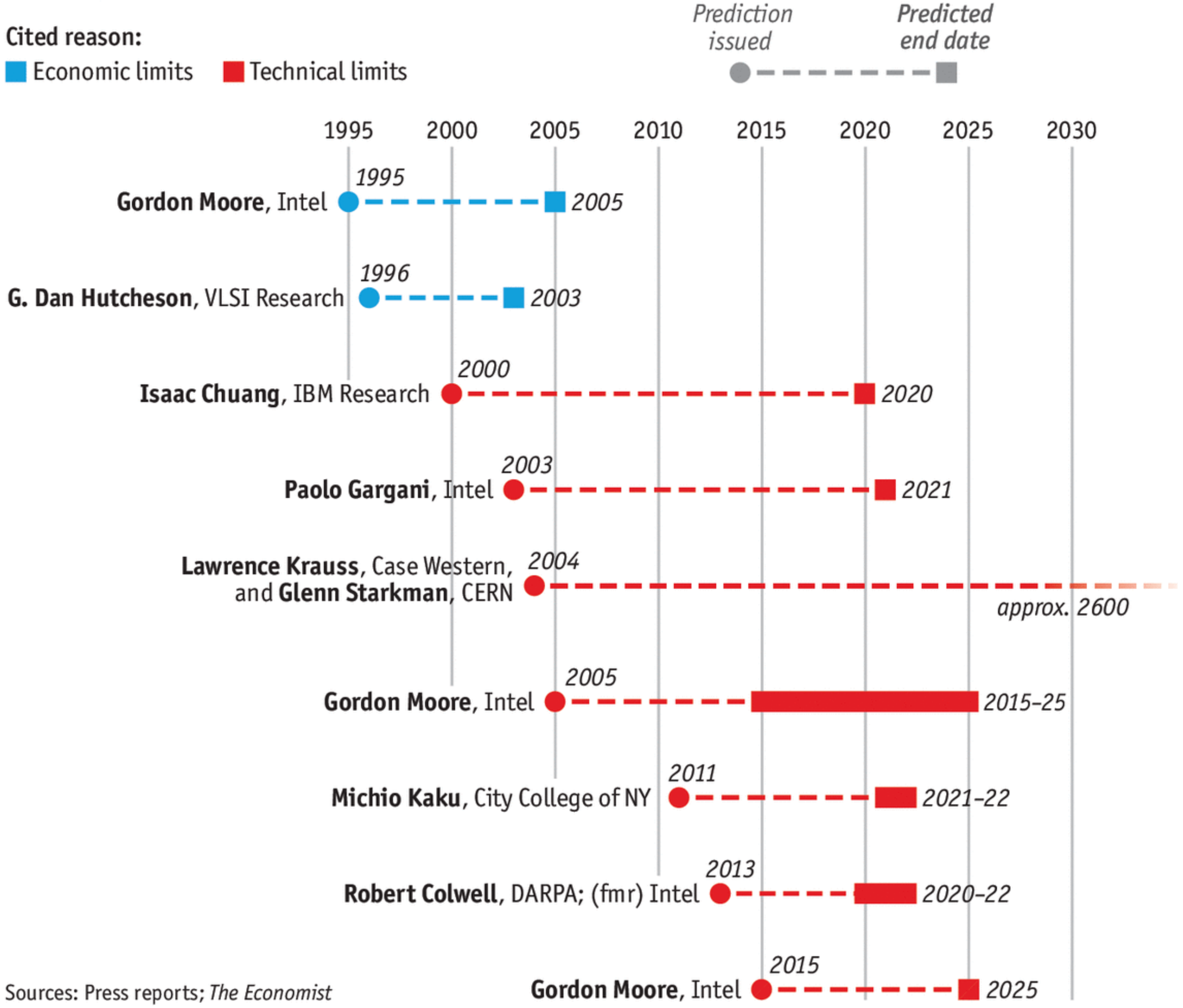
Selected predictions for the end of Moore's Law



Sources: Press reports; *The Economist*

Economist.com

Selected predictions for the end of Moore's Law

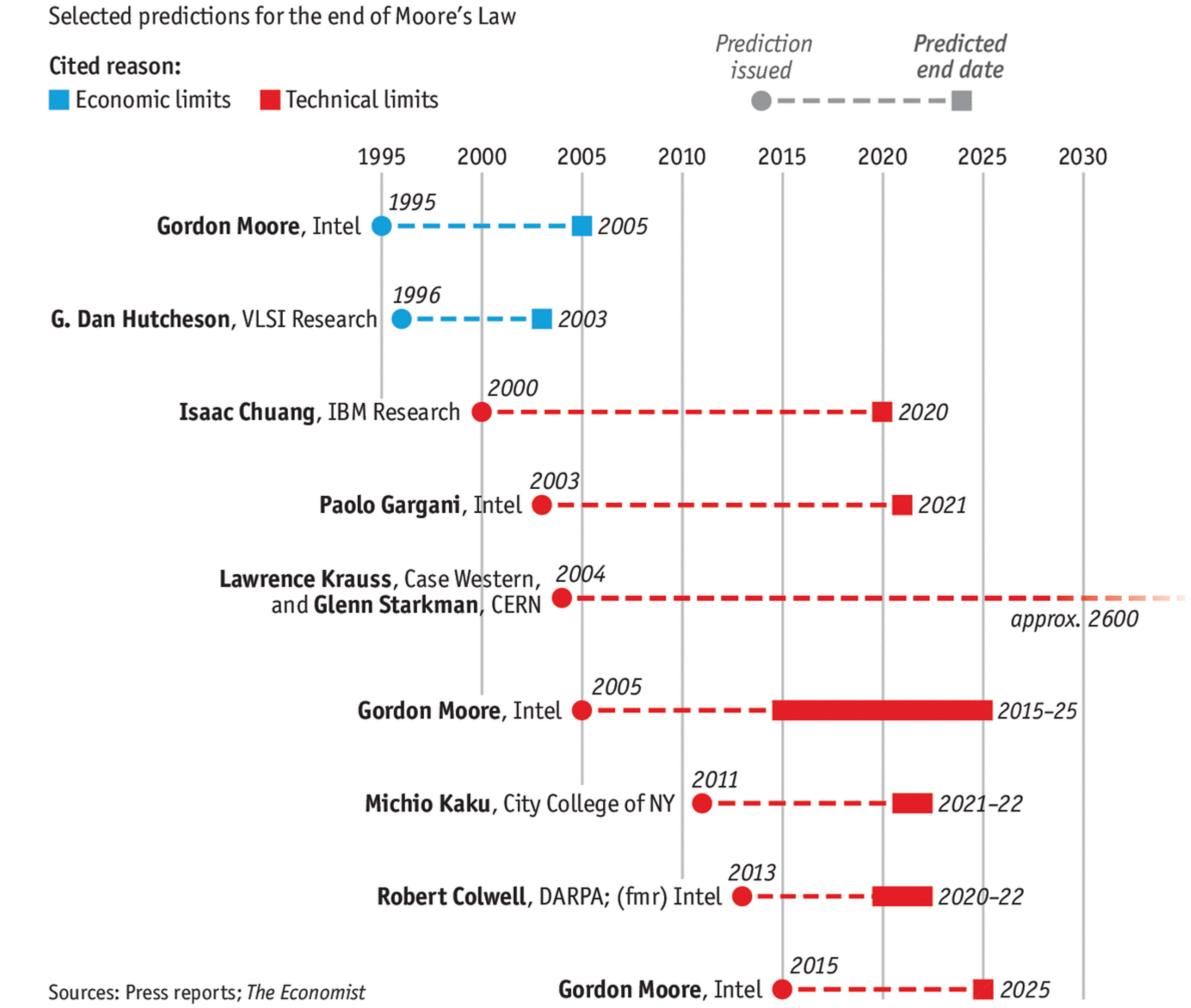


Sources: Press reports; *The Economist*

Economist.com

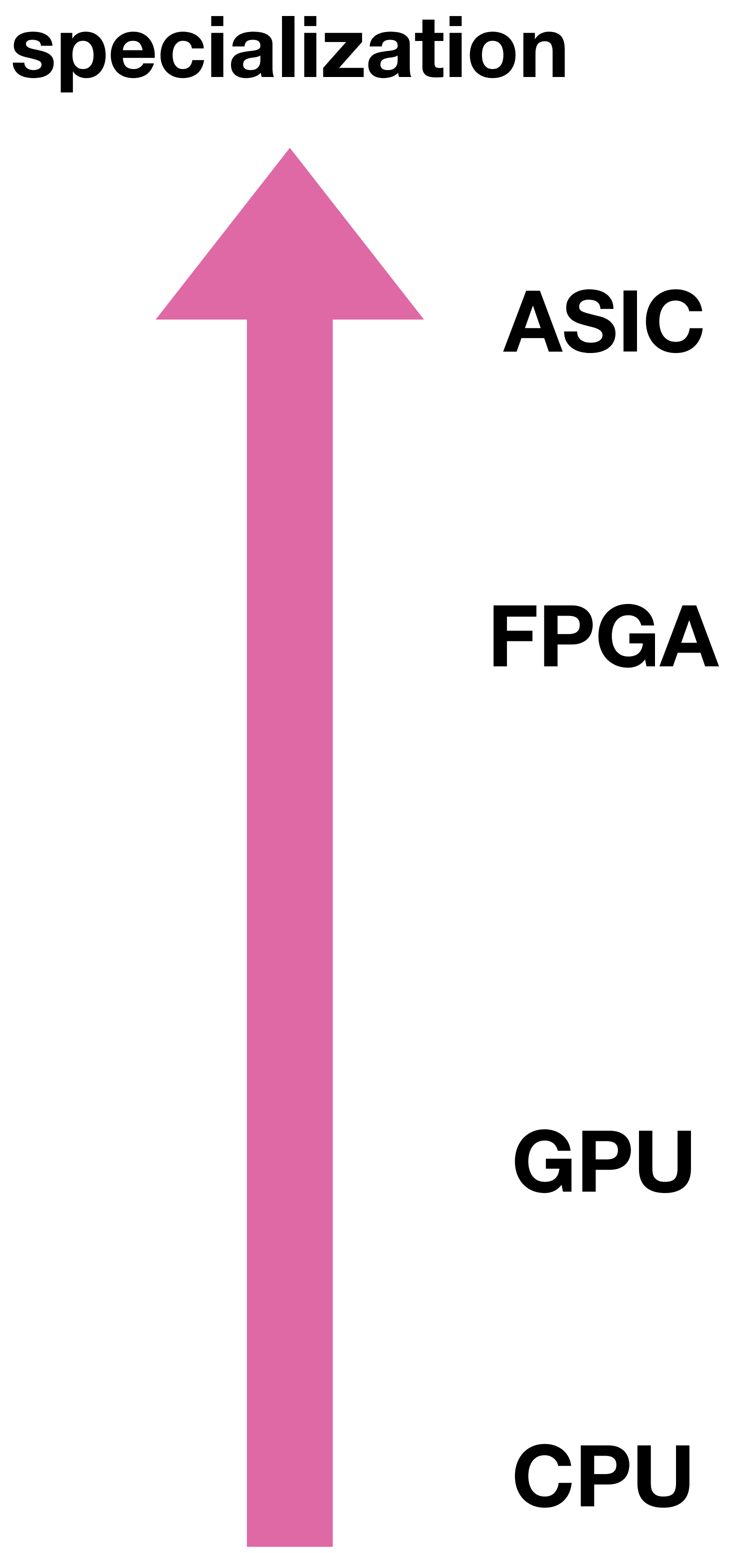
motivation for new, specialized architectures/accelerators

motivation for new, specialized architectures/accelerators



Sources: Press reports; *The Economist*

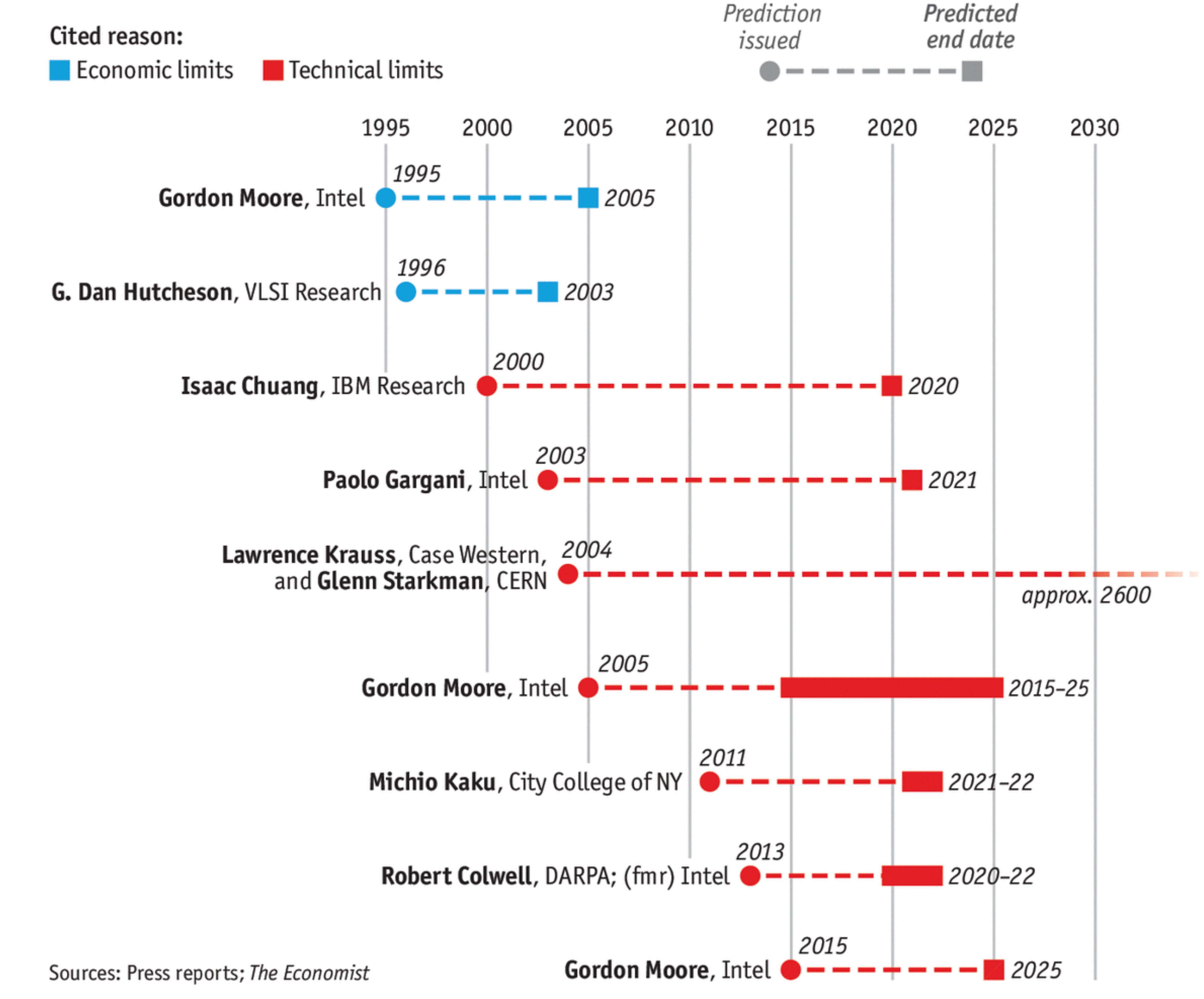
Economist.com



acronym soup!

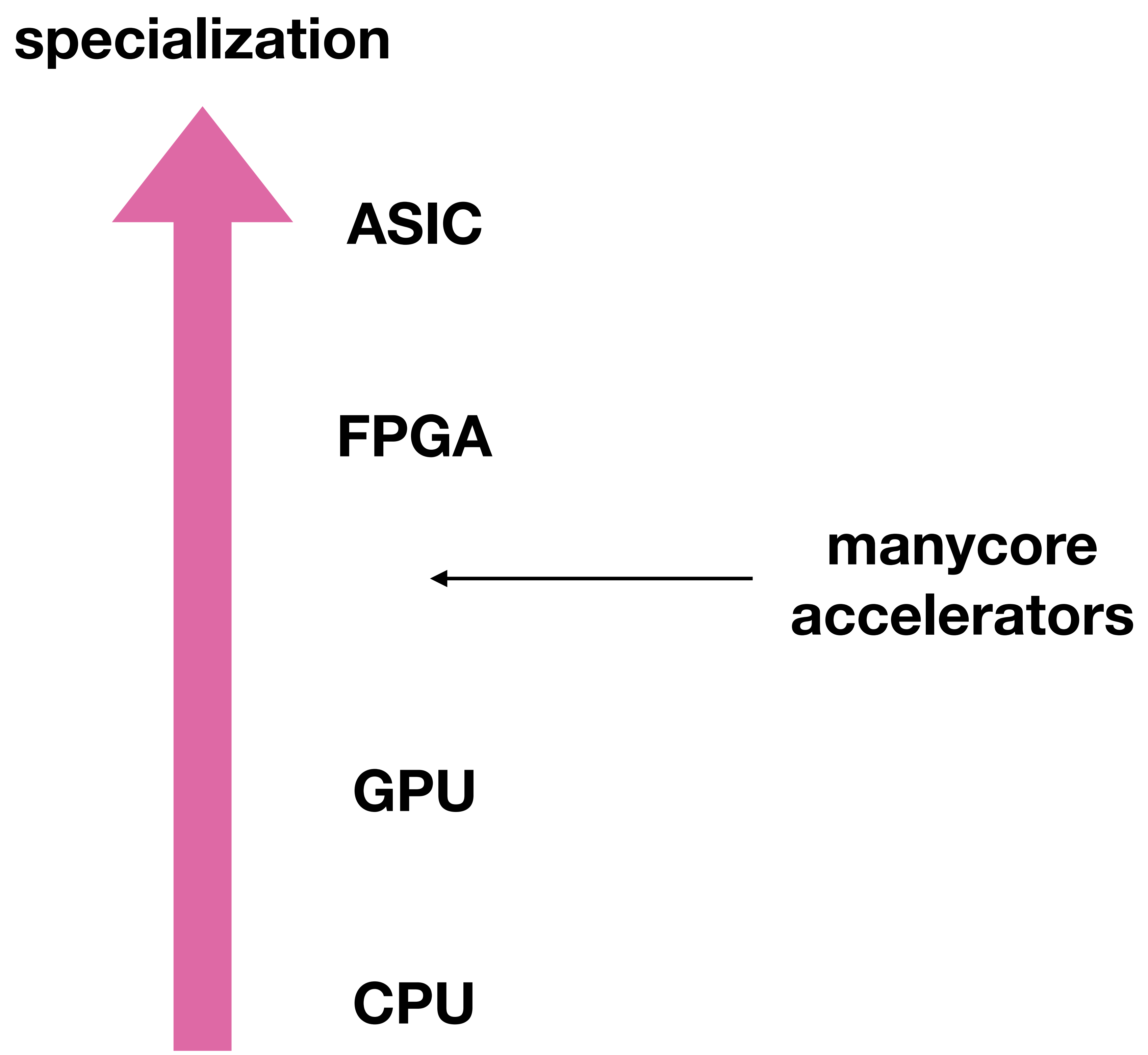
motivation for new, specialized architectures/accelerators

Selected predictions for the end of Moore's Law



Sources: Press reports; *The Economist*

Economist.com



specialization

ASIC

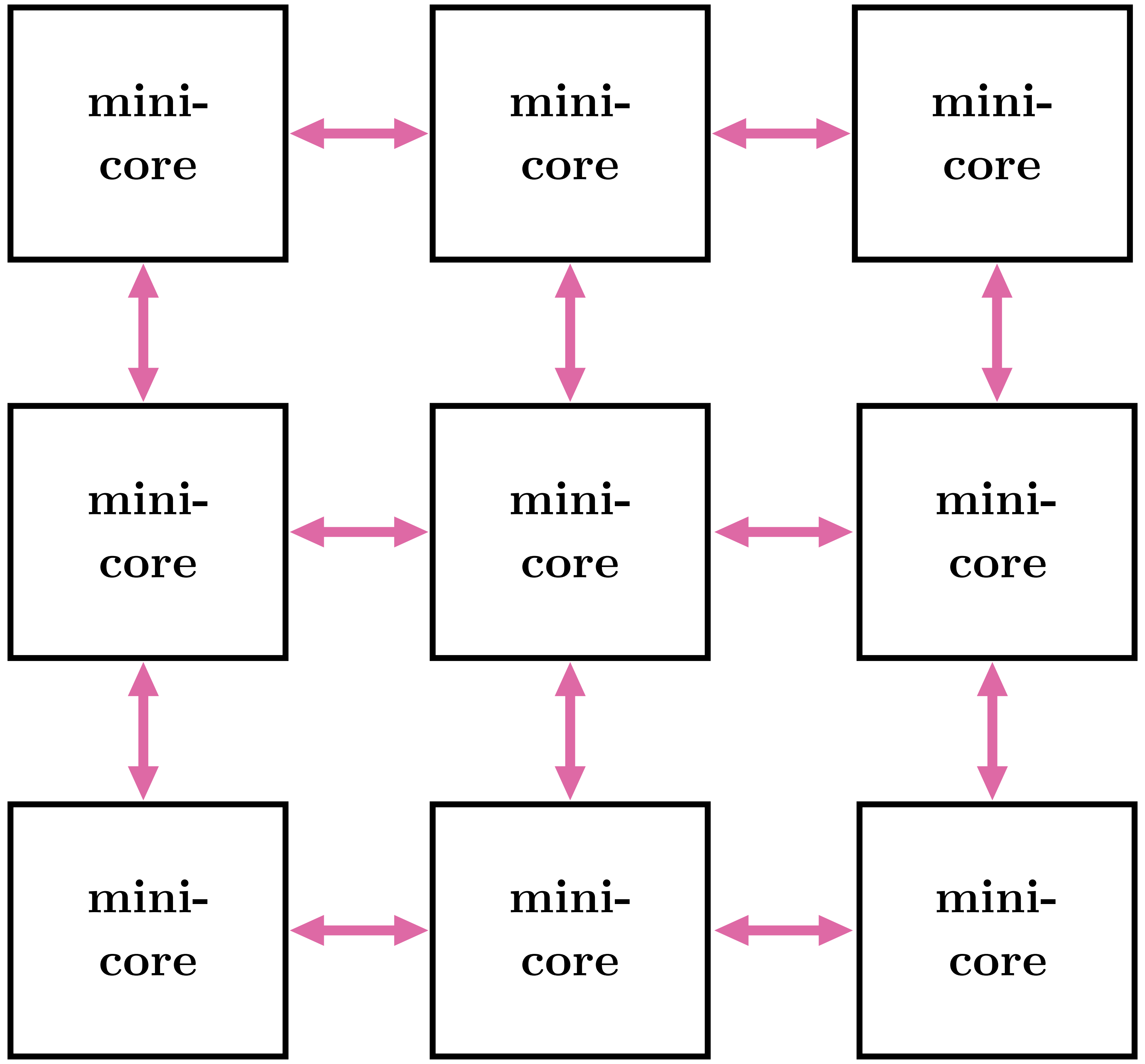
FPGA

manycore accelerators

GPU

CPU

manycore accelerators

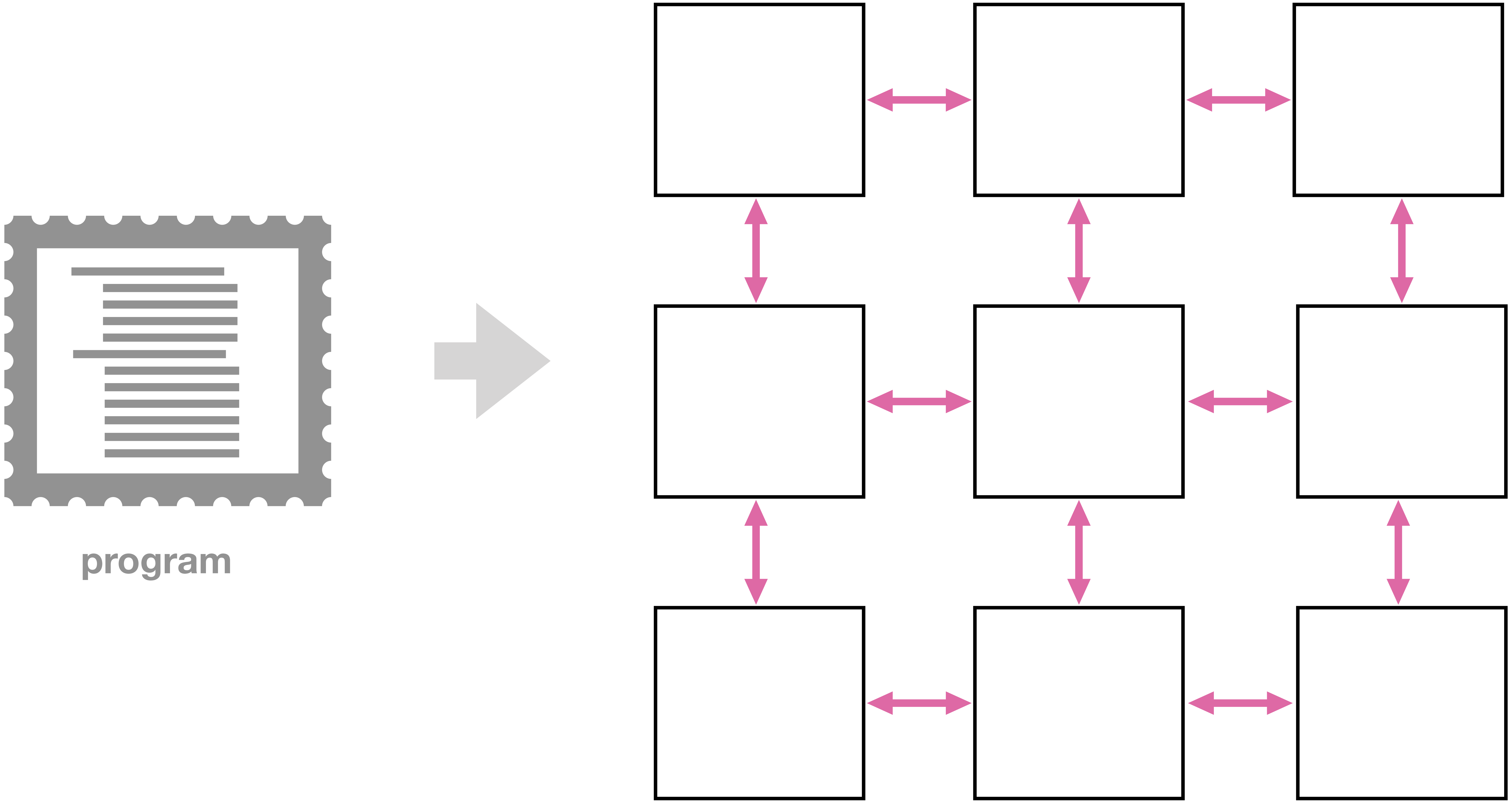


- differ from “multicore” processors
- combine many small, simple processor cores in a spatial layout
- communication via nearest neighbors
- programmer or compiler must specify partitioning

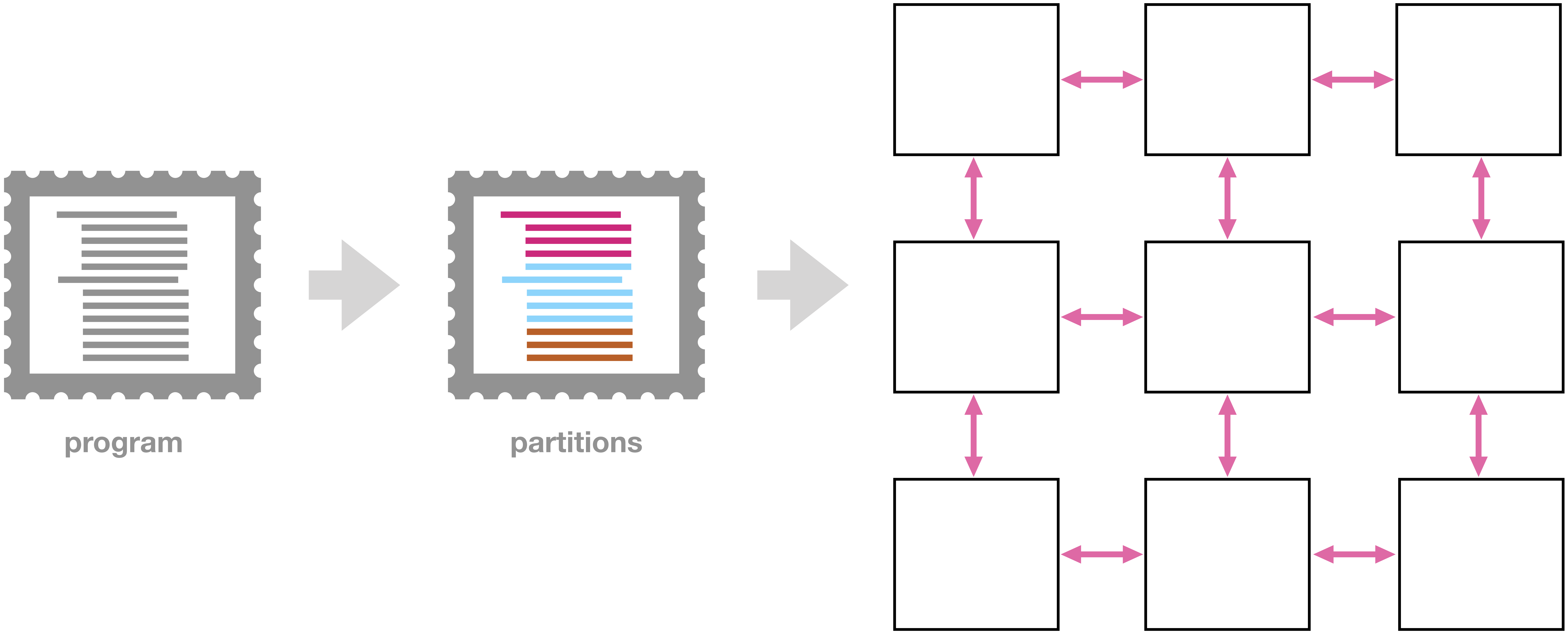
the challenge:

write programs at a higher level that leverage the spatial layout

partitioning



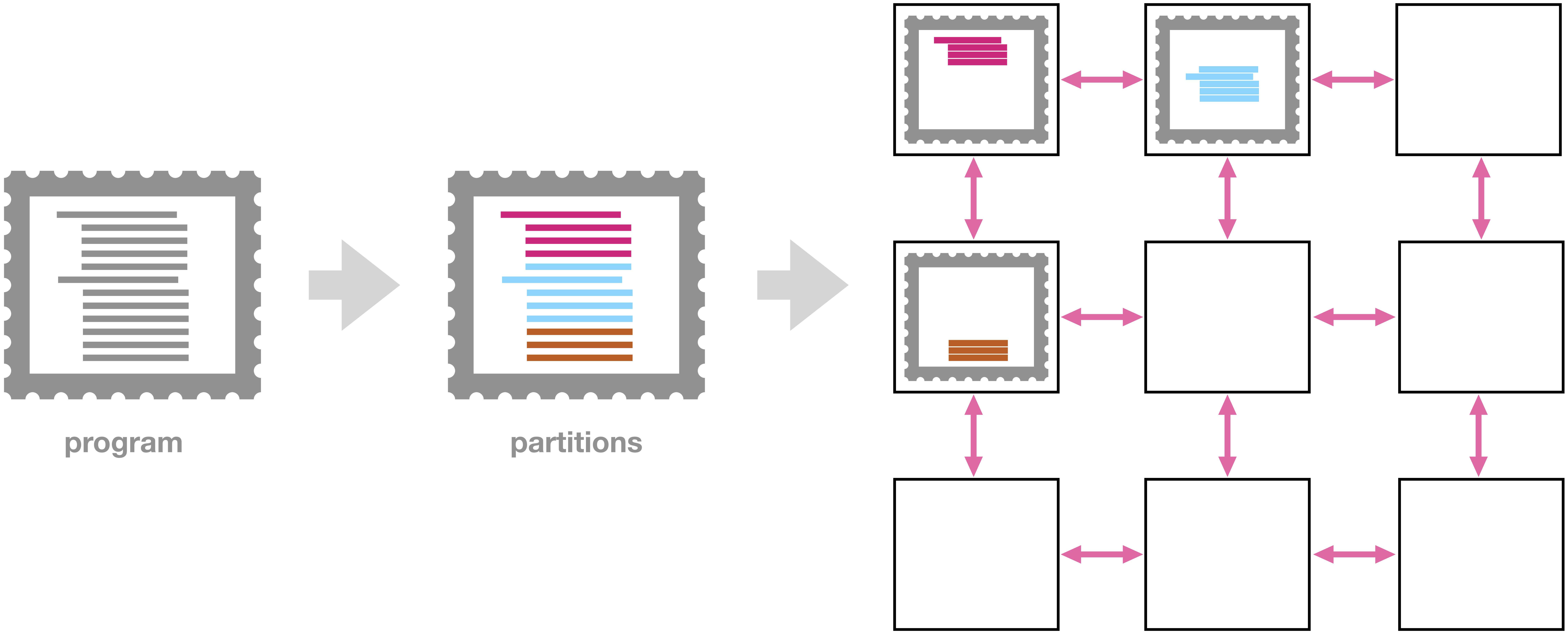
partitioning



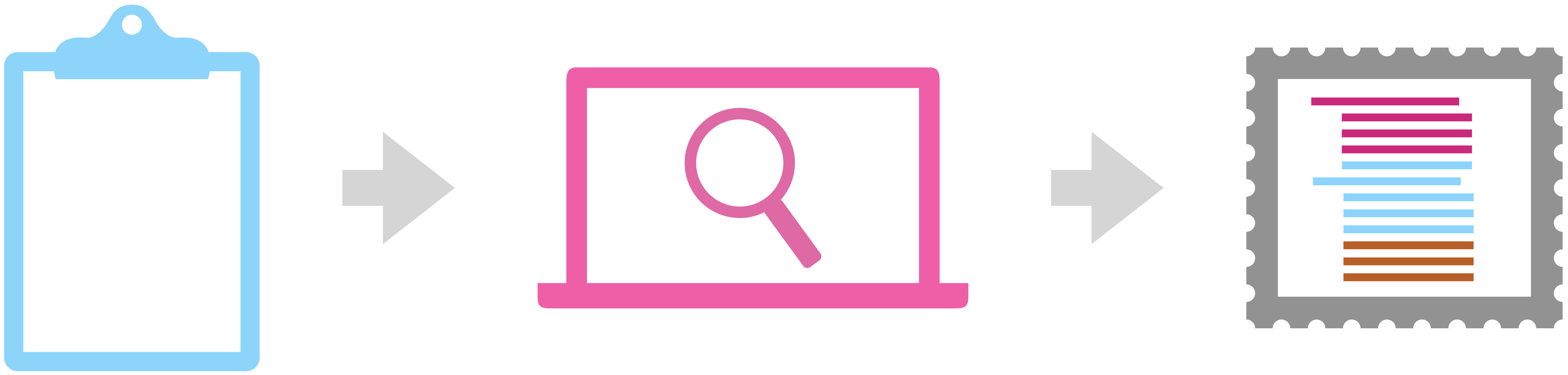
program

partitions

partitioning



program synthesis

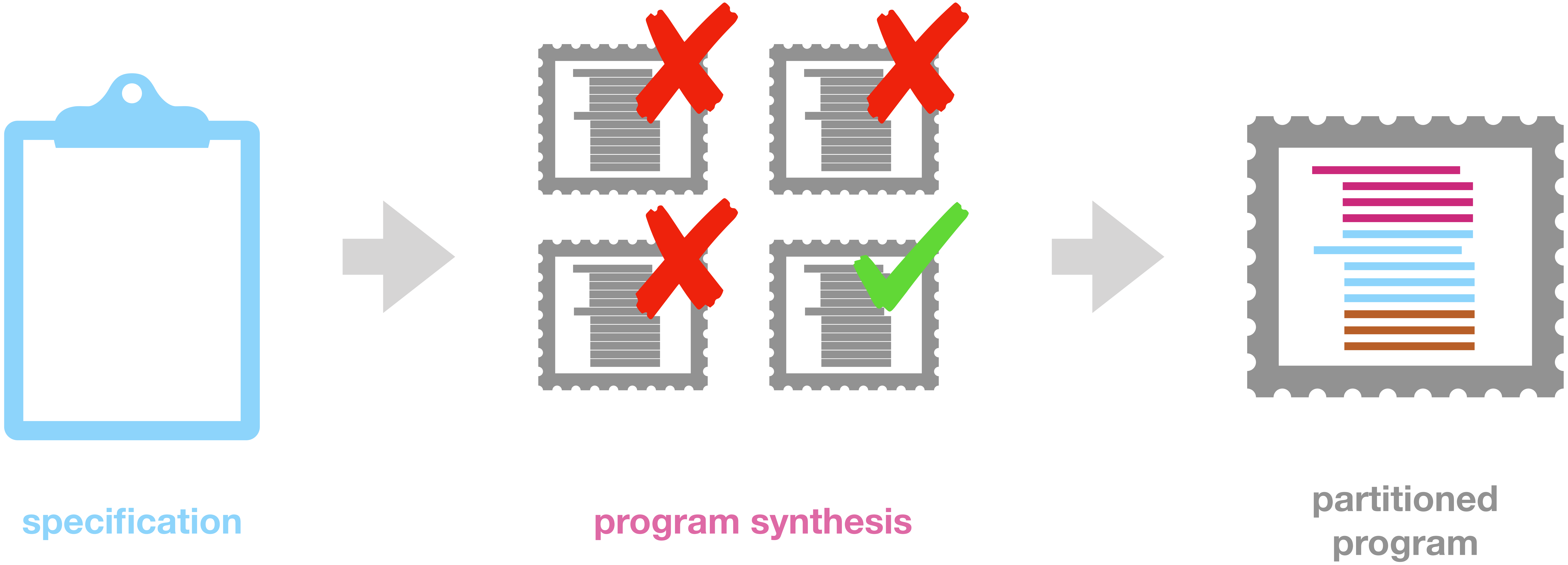


specification

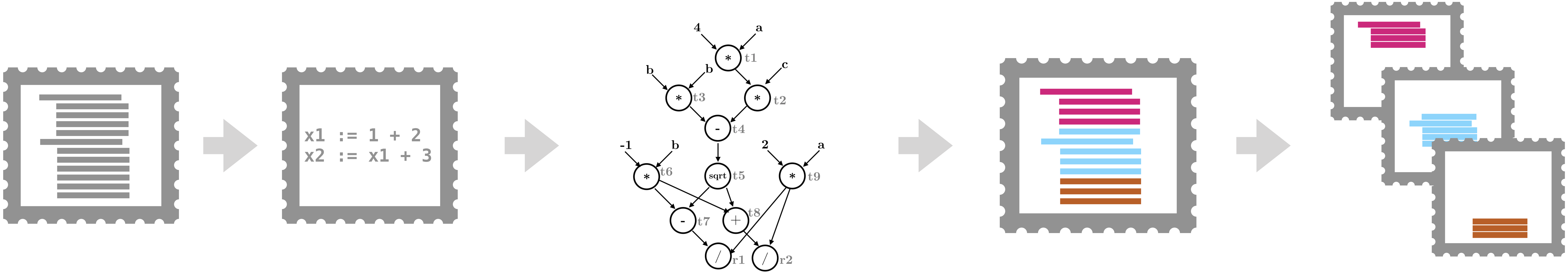
program synthesis

partitioned
program

program synthesis



compiler overview



```
x1 := 1 + 2
x2 := x1 + 3
```

high level program

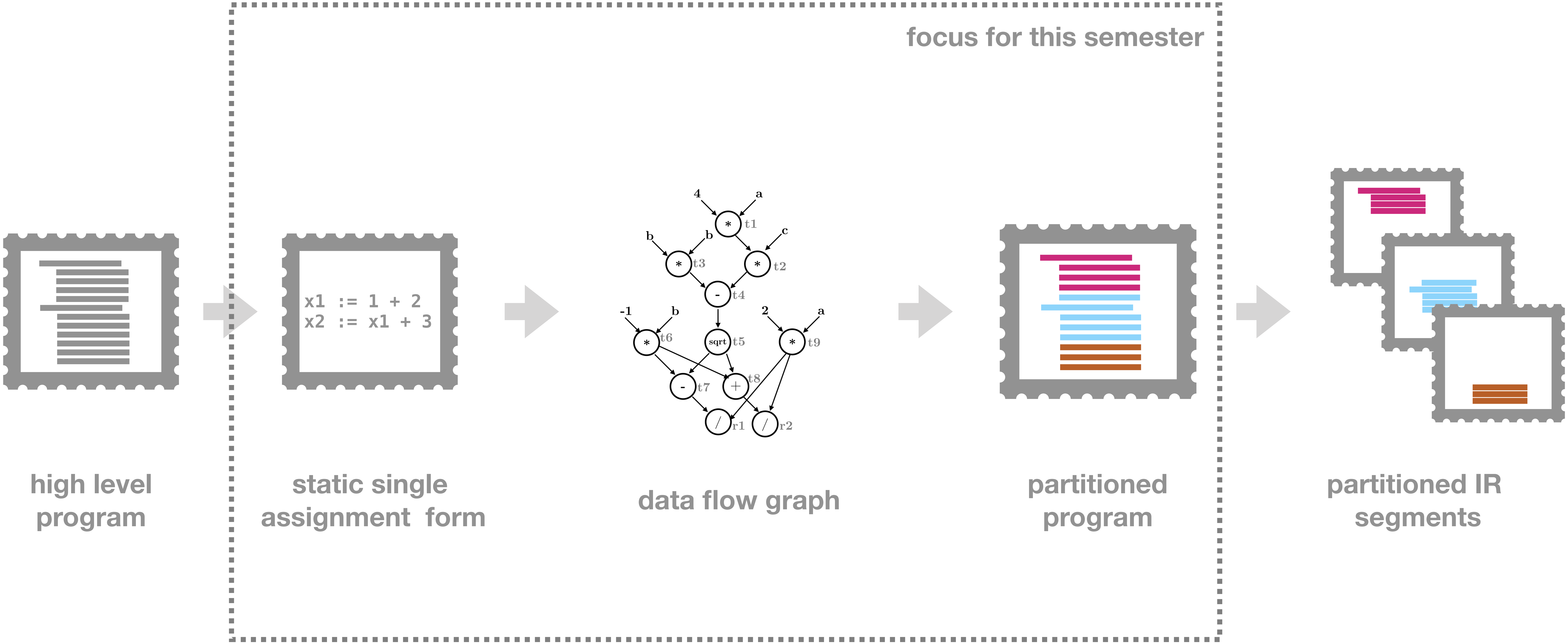
static single assignment form

data flow graph

partitioned program

partitioned IR segments

compiler overview



focus for this semester

```
x1 := 1 + 2
x2 := x1 + 3
```

high level program

static single assignment form

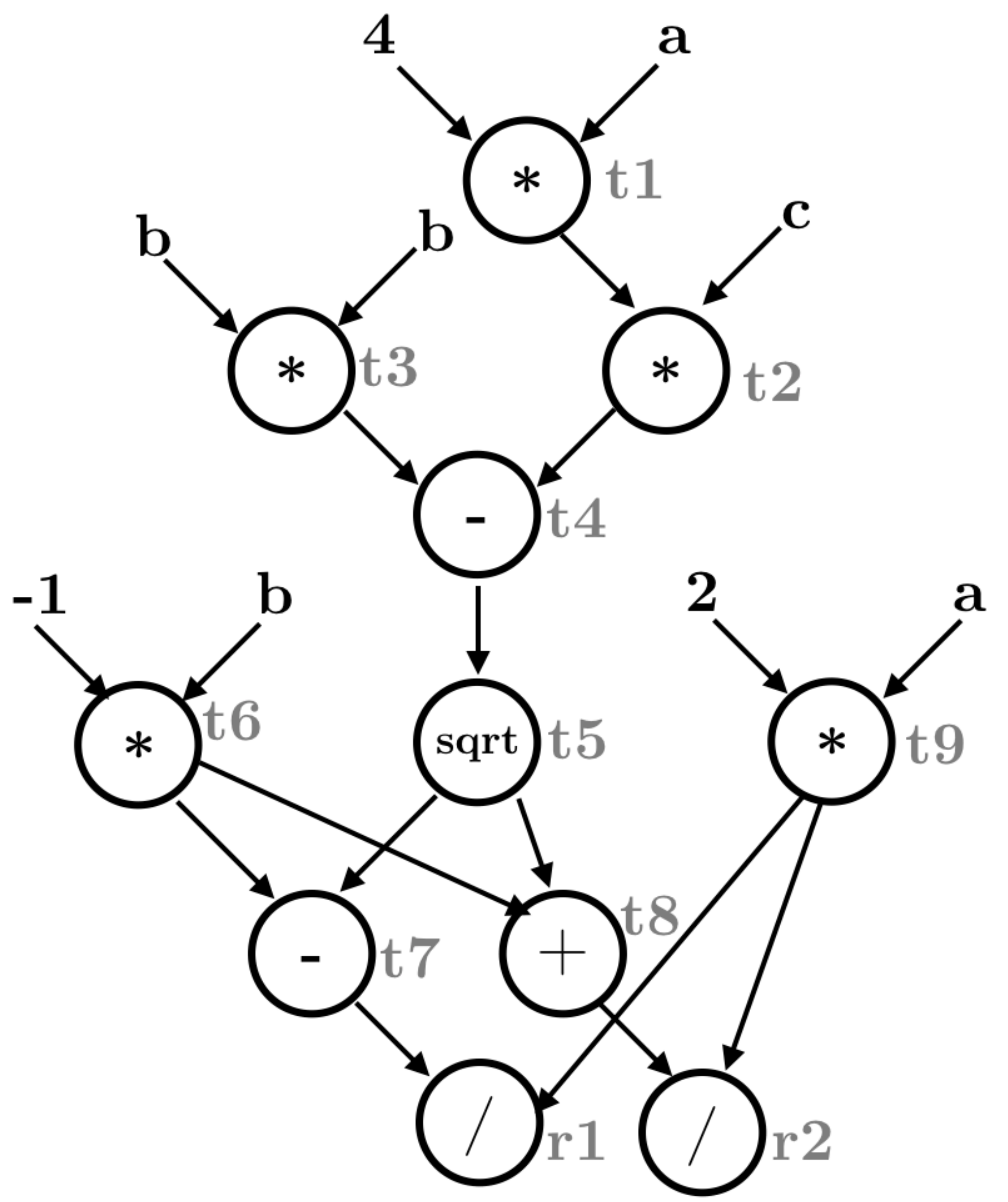
data flow graph

partitioned program

partitioned IR segments

constraint generation

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

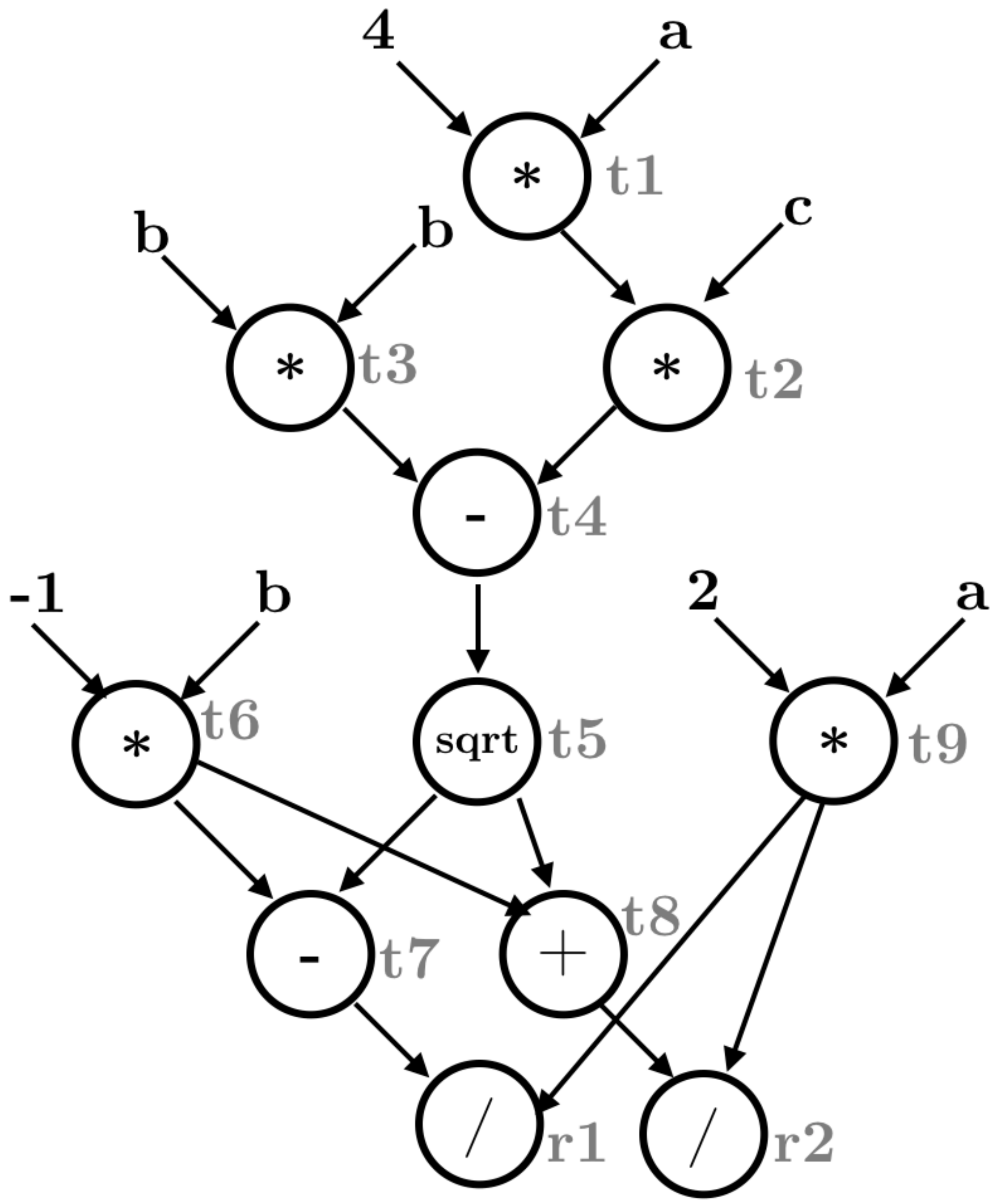


- track partition and start/end times `(p, (t1, t2))` per operation

constraint generation

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

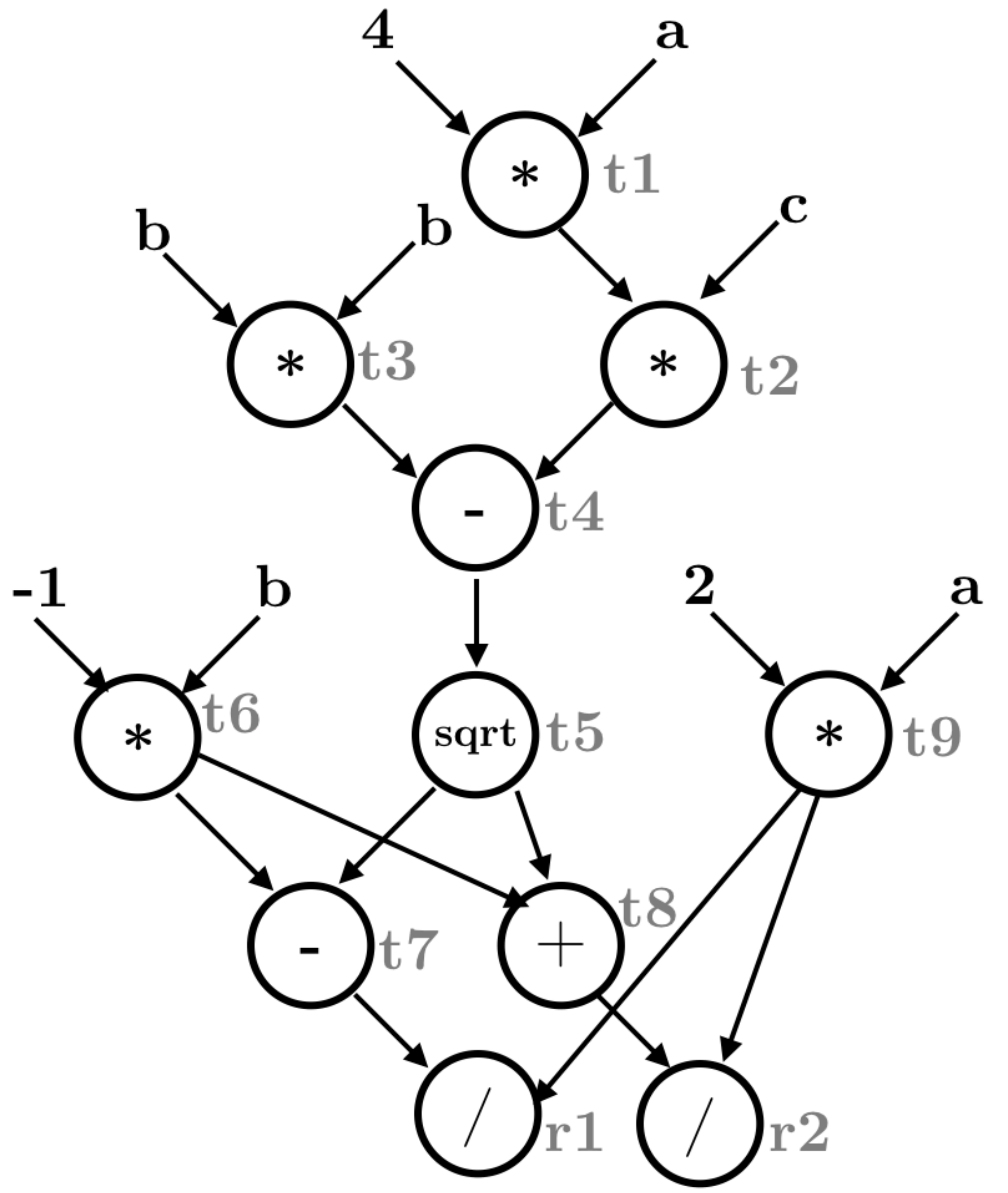
- track partition and start/end times `(p, (t1, t2))` per operation



time example: no 2 operations on the same core at the same time

constraint generation

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$



- track partition and start/end times `(p, (t1, t2))` per operation

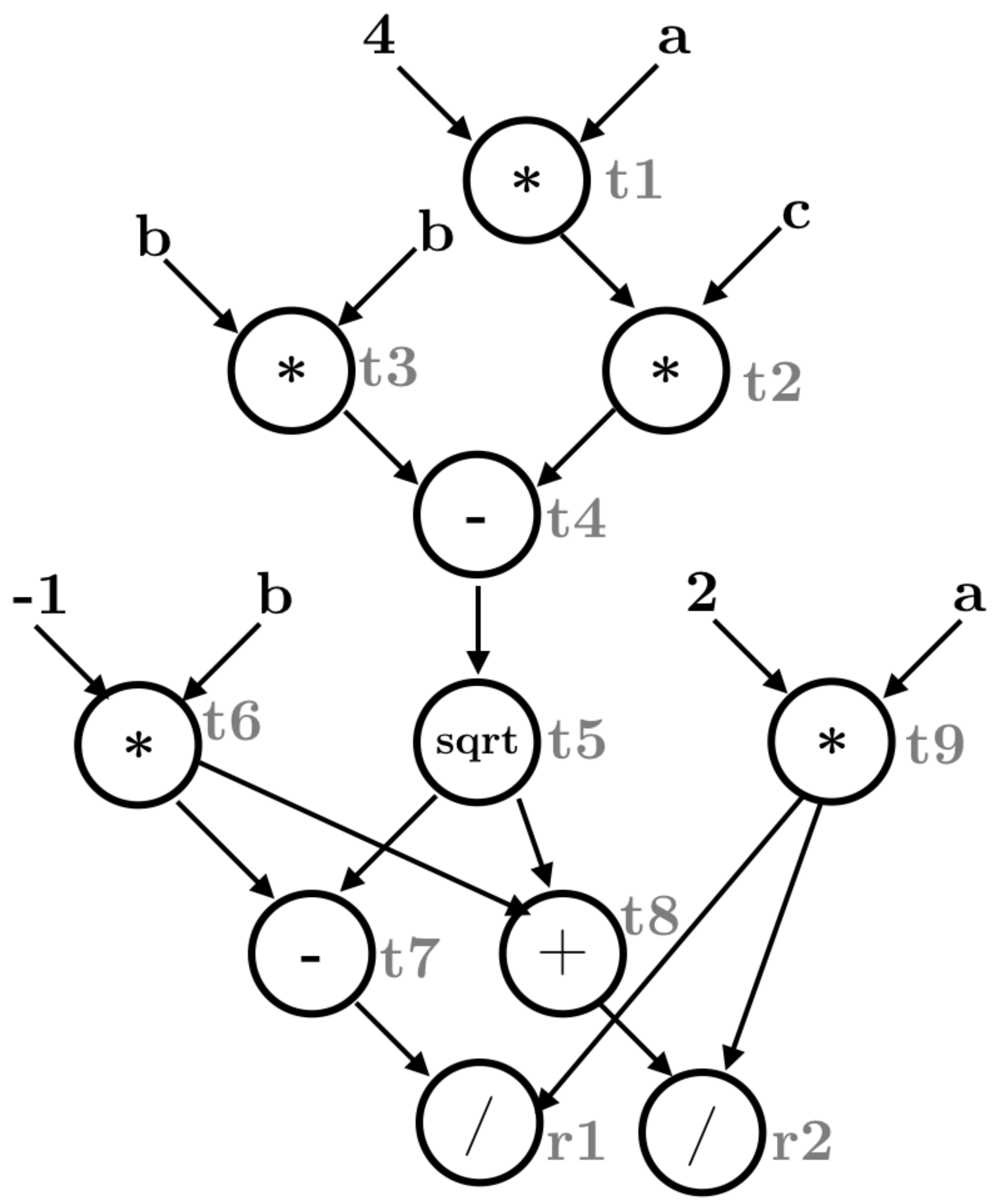
time example: no 2 operations on the same core at the same time

```
∀ disj. n, n’.

(p != p’) => (t2 < t1’) ∨ (t1 > t2’)
```

constraint generation

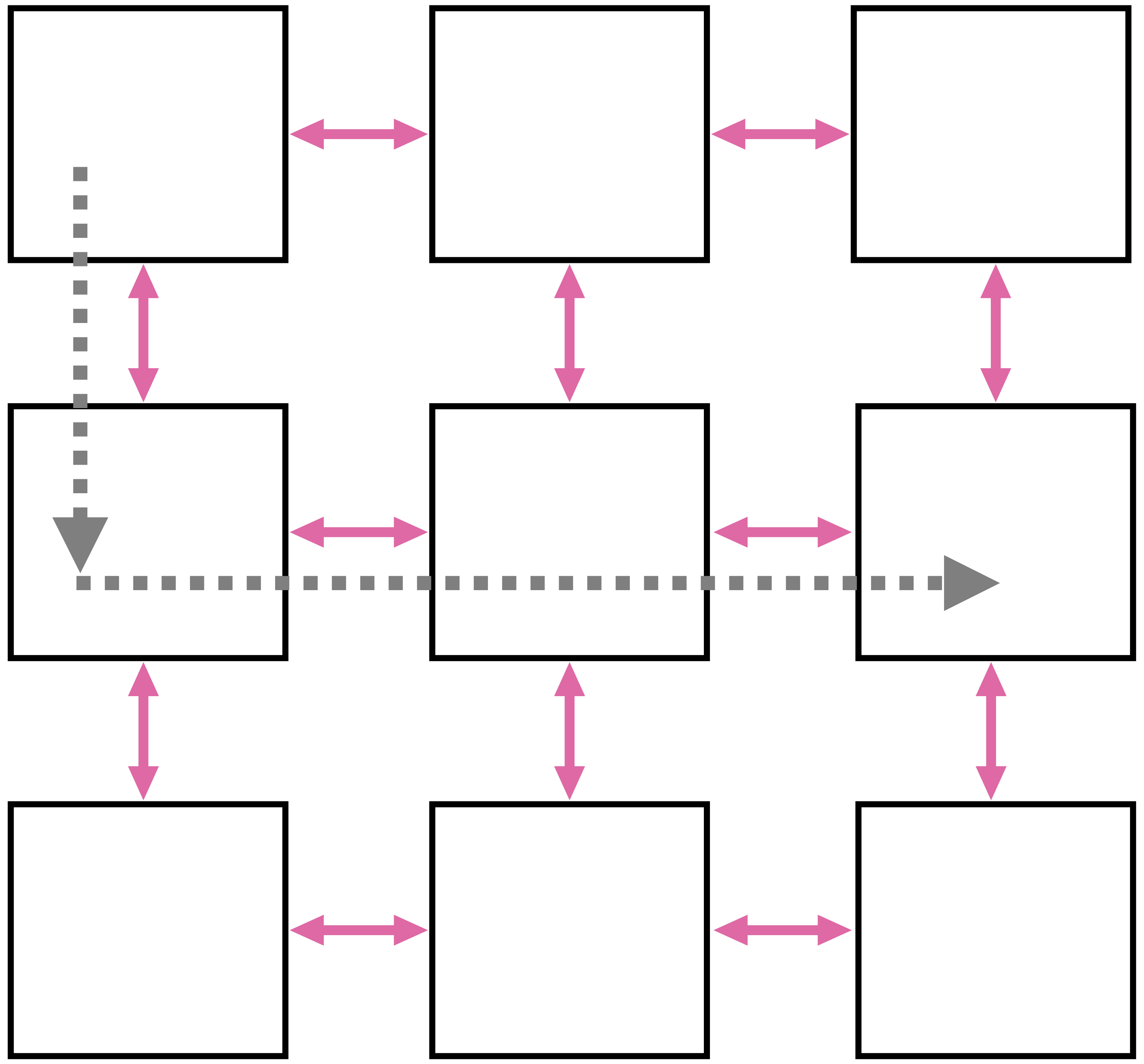
$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$



- track partition and start/end times `(p, (t1, t2))` per operation

communication example: delay between partitions is the Manhattan distance (number of hops)

constraint generation



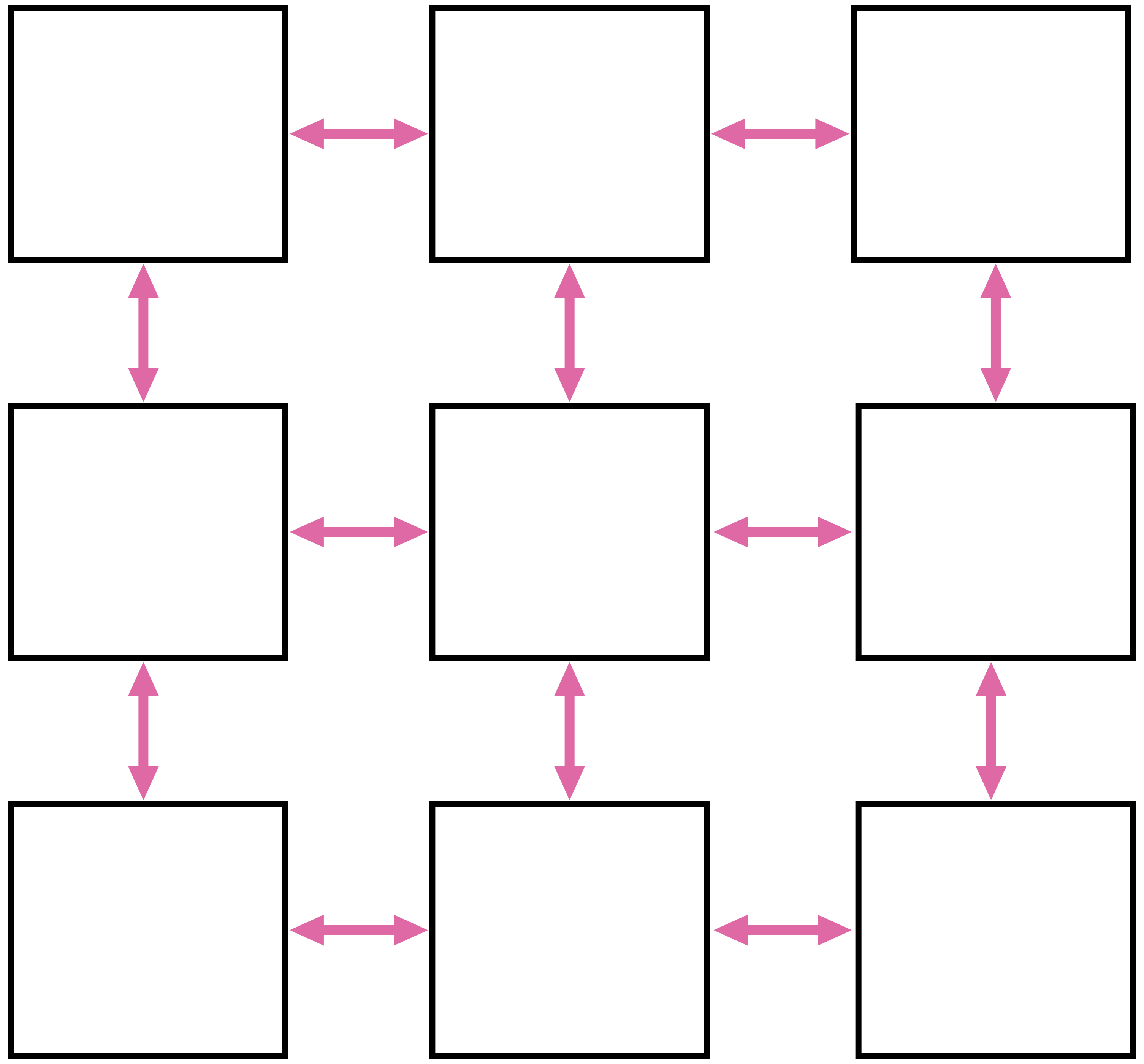
- track partition and start/end times `(p, (t1, t2))` per operation

communication example: delay between partitions is the Manhattan distance (number of hops)

```
∀ edges n, n’. t1’ >= t2 + dist(p, p’)
```

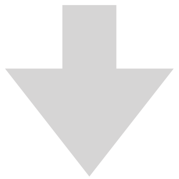
```
dist = abs(x − x') + abs(y − y')
```

constraint generation



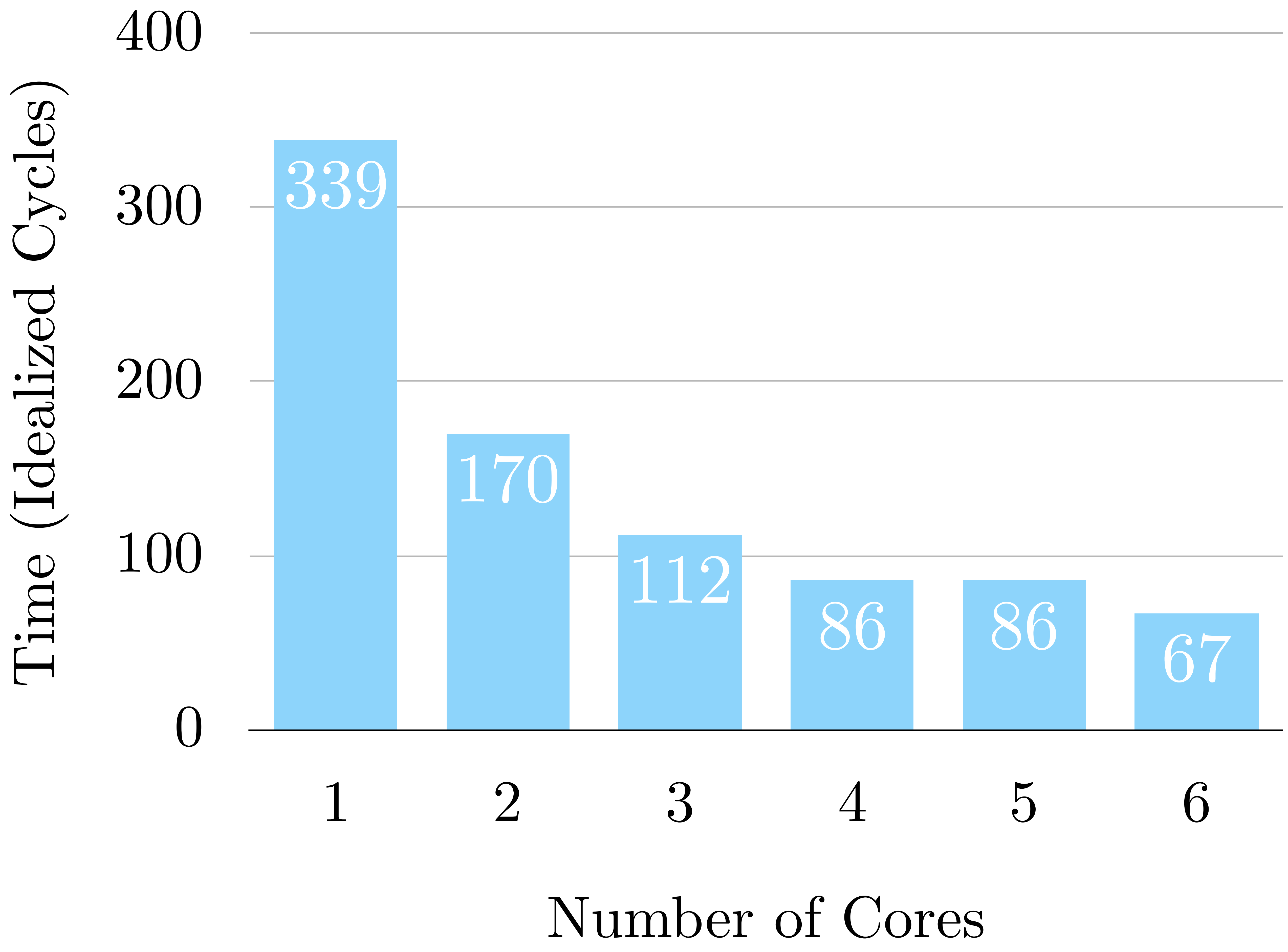
- track partition and start/end times `(p, (t1, t2))` per operation

final goal: find an assignment where the latest time is <= the goal



iterative solution starting at the sequential upper-bound

evaluation



- goal: run IR fragments on a cycle-level simulator
- current: compare the model's time (idealized cycles) across core configurations
- shown: 50 LOC program, idealized upper bound 339 cycles