ULB UNIVERSITÉ LIBRE DE BRUXELLES

HIPPEROS
Predictable Real-Time, Proven Performance

SOUTENANCE THESE

Antonio Paolillo

17 octobre 2018

Agenda

1. Domaine, problème, intuition
2. Contributions et thèse
3. Directions futures

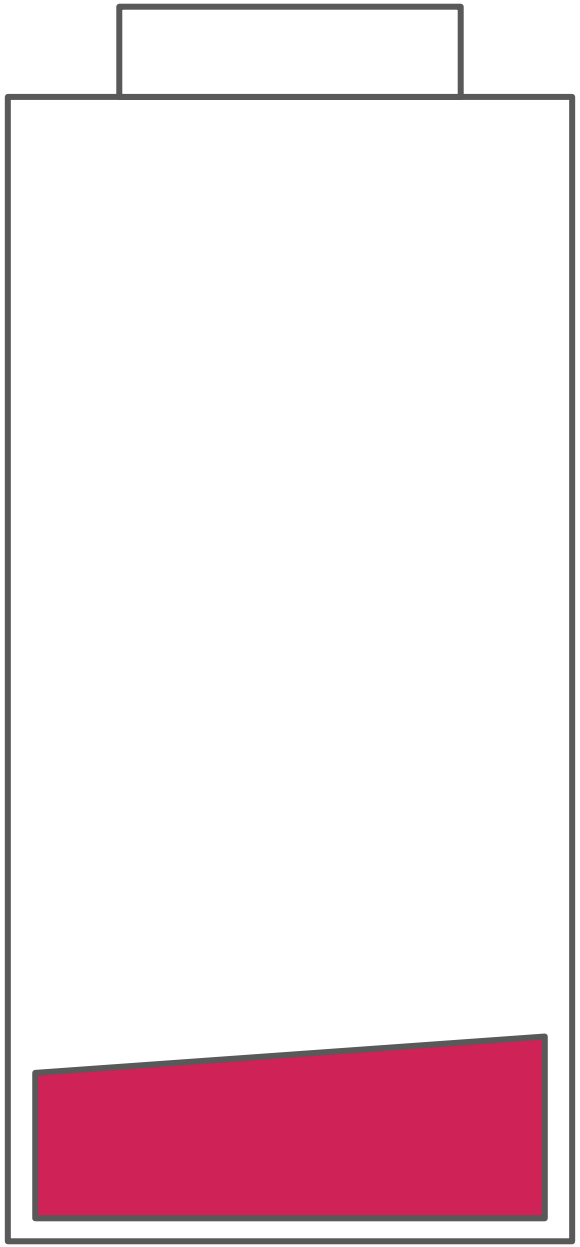
Agenda

1. **Domaine, problème, intuition**
2. Contributions et thèse
3. Directions futures

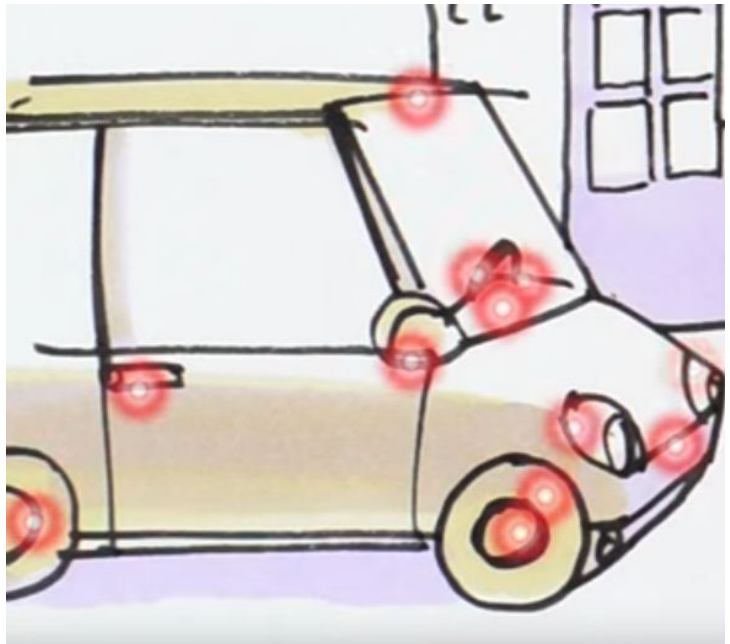

TV


TV


TV

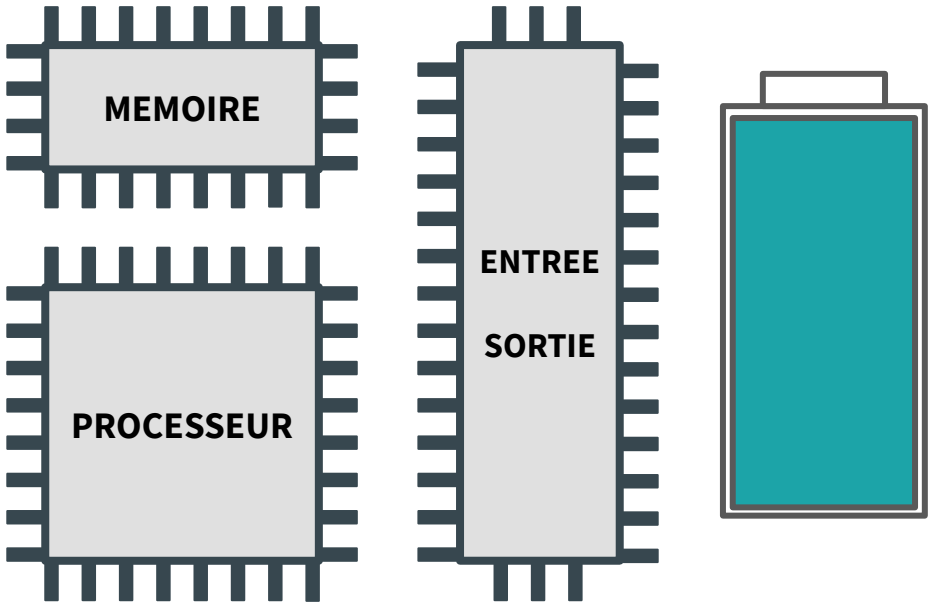


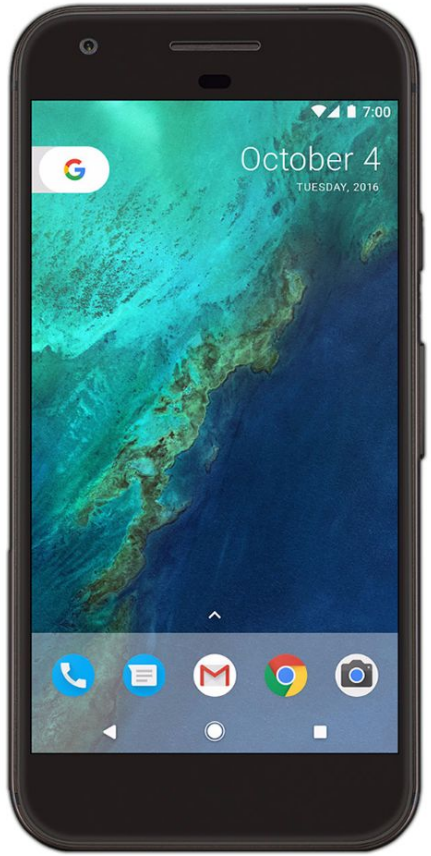

boundary devices
SABER LITE

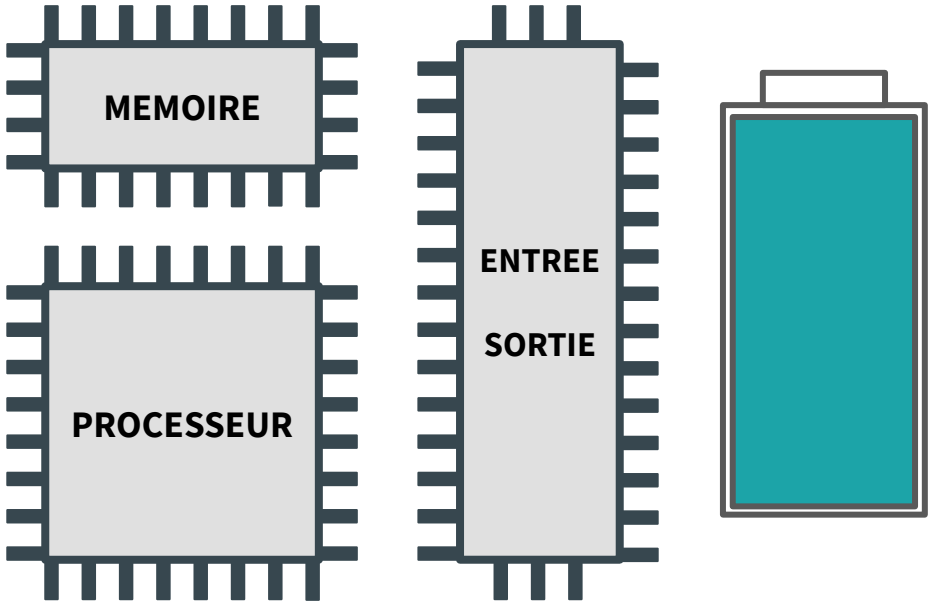


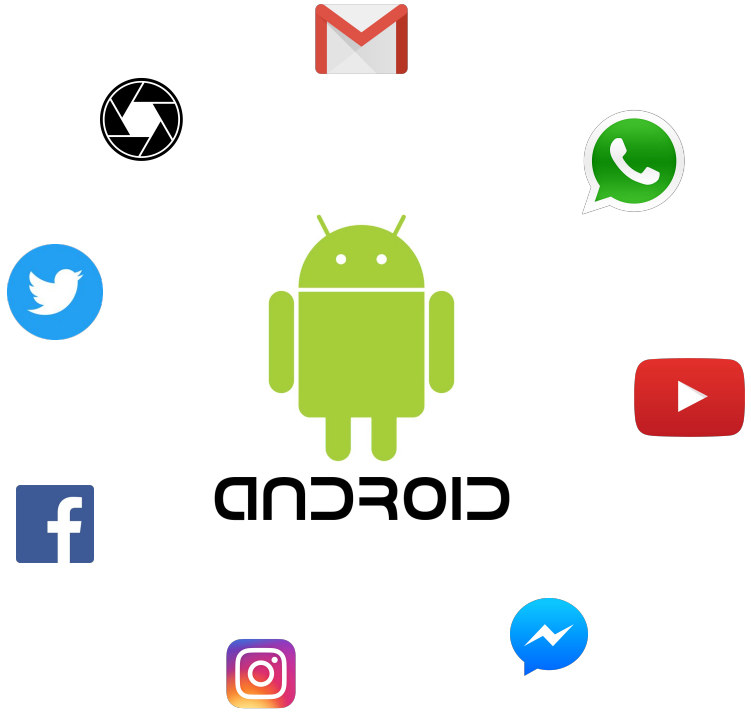

Boundary Devices
SABER LITE

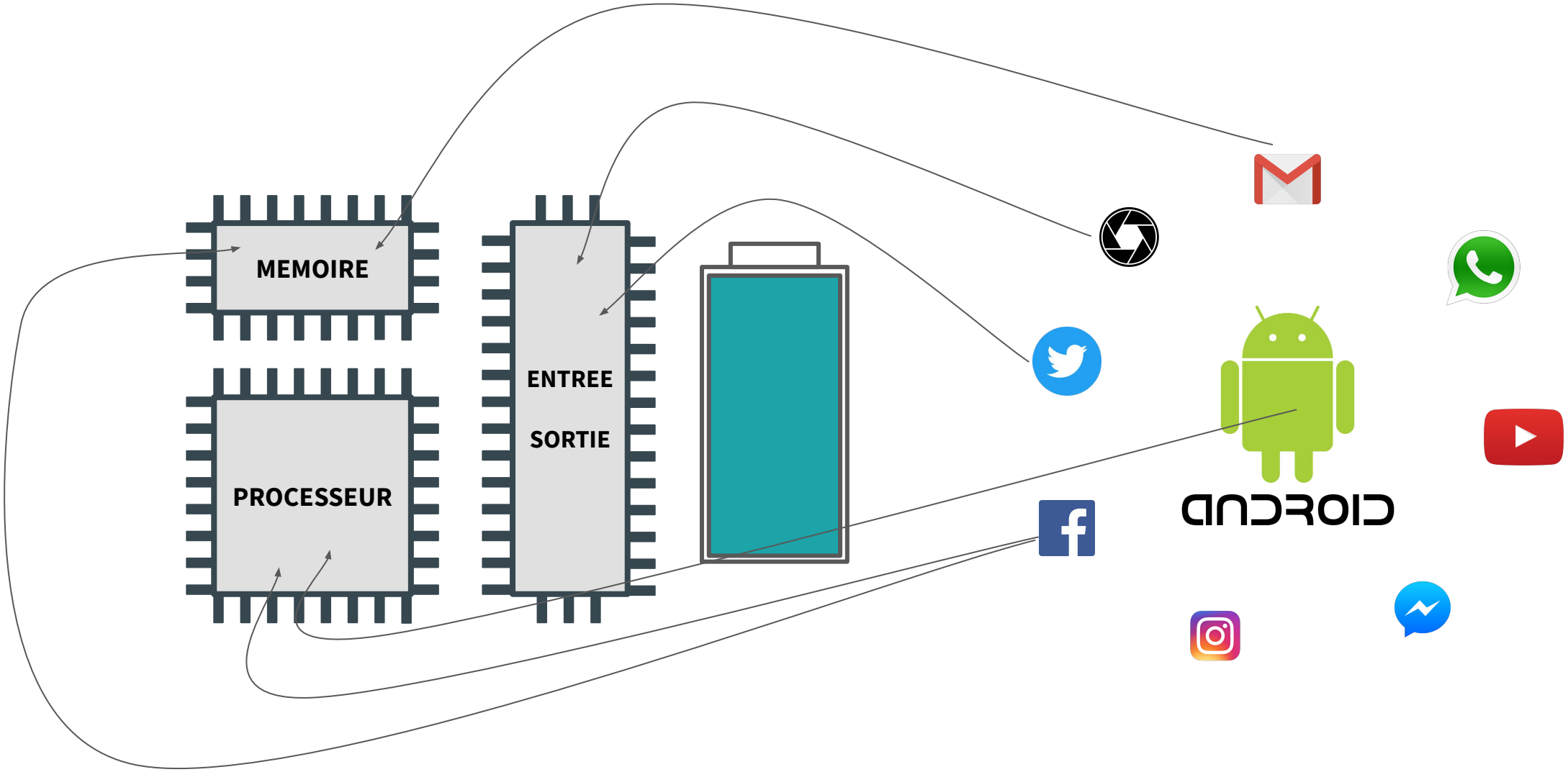

7:00
October 4
TUESDAY, 2016

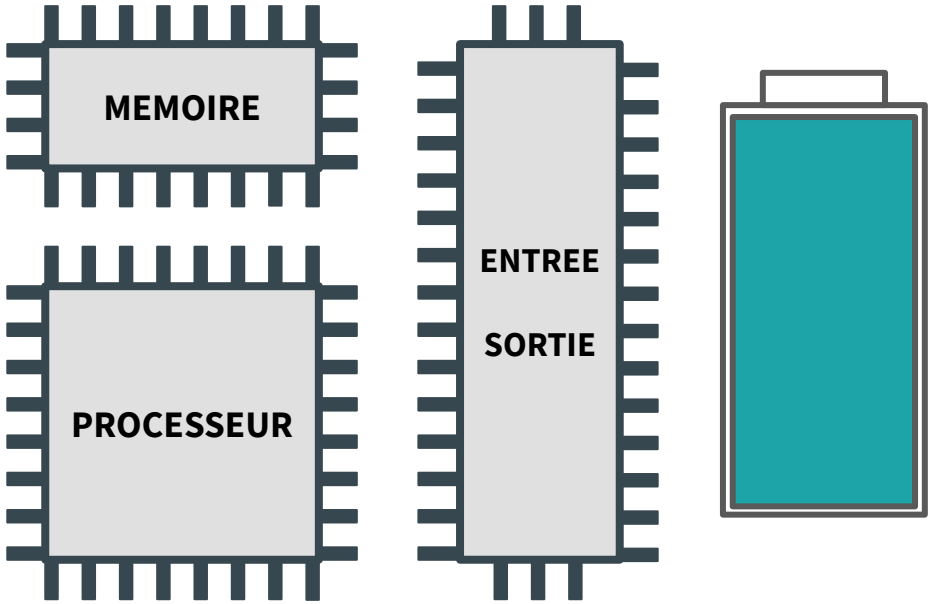

MEMOIRE
PROCESSEUR
ENTREE
SORTIE

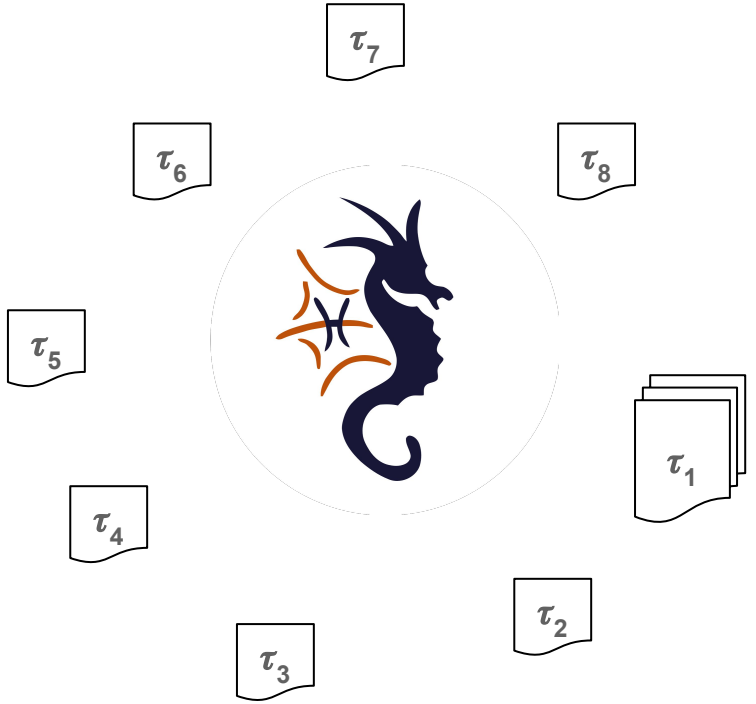

7:00
October 4
TUESDAY, 2016

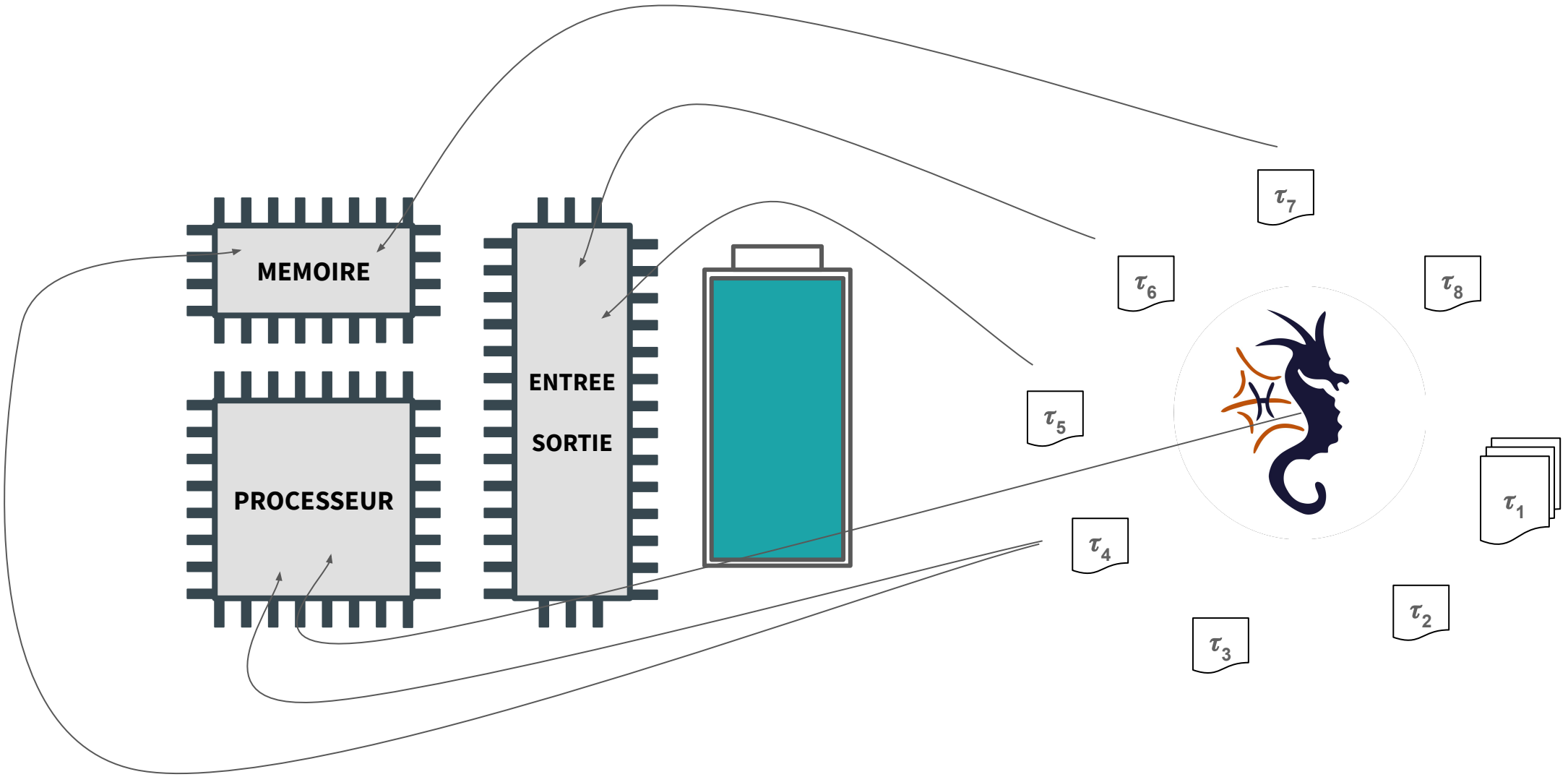

MEMOIRE
PROCESSEUR
ENTREE
SORTIE

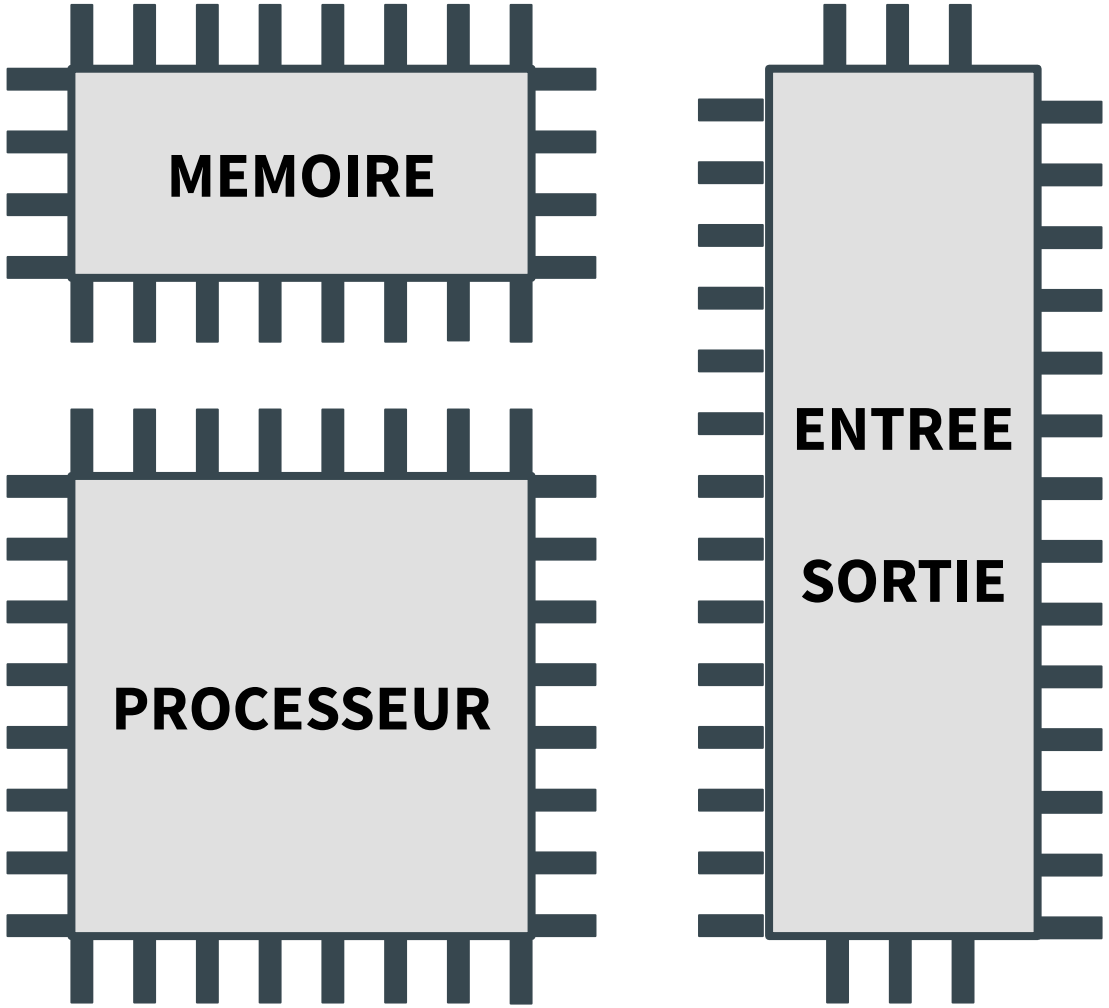

ANDROID

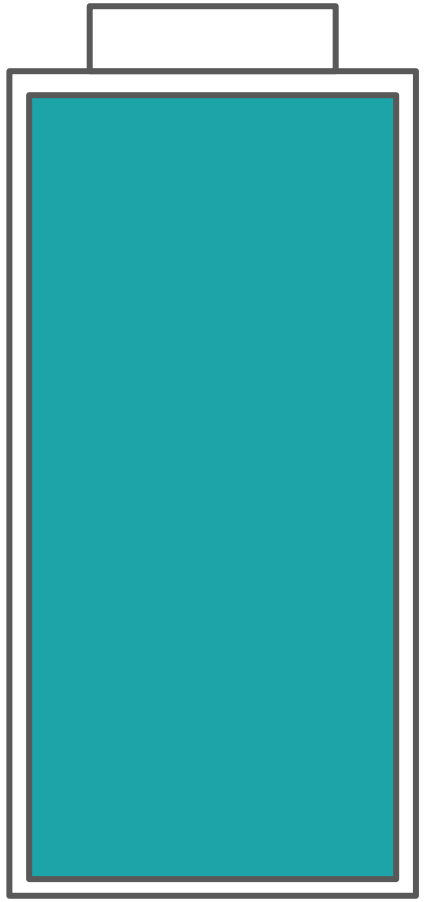

MEMOIRE
PROCESSEUR
ENTREE
SORTIE
ANDROID

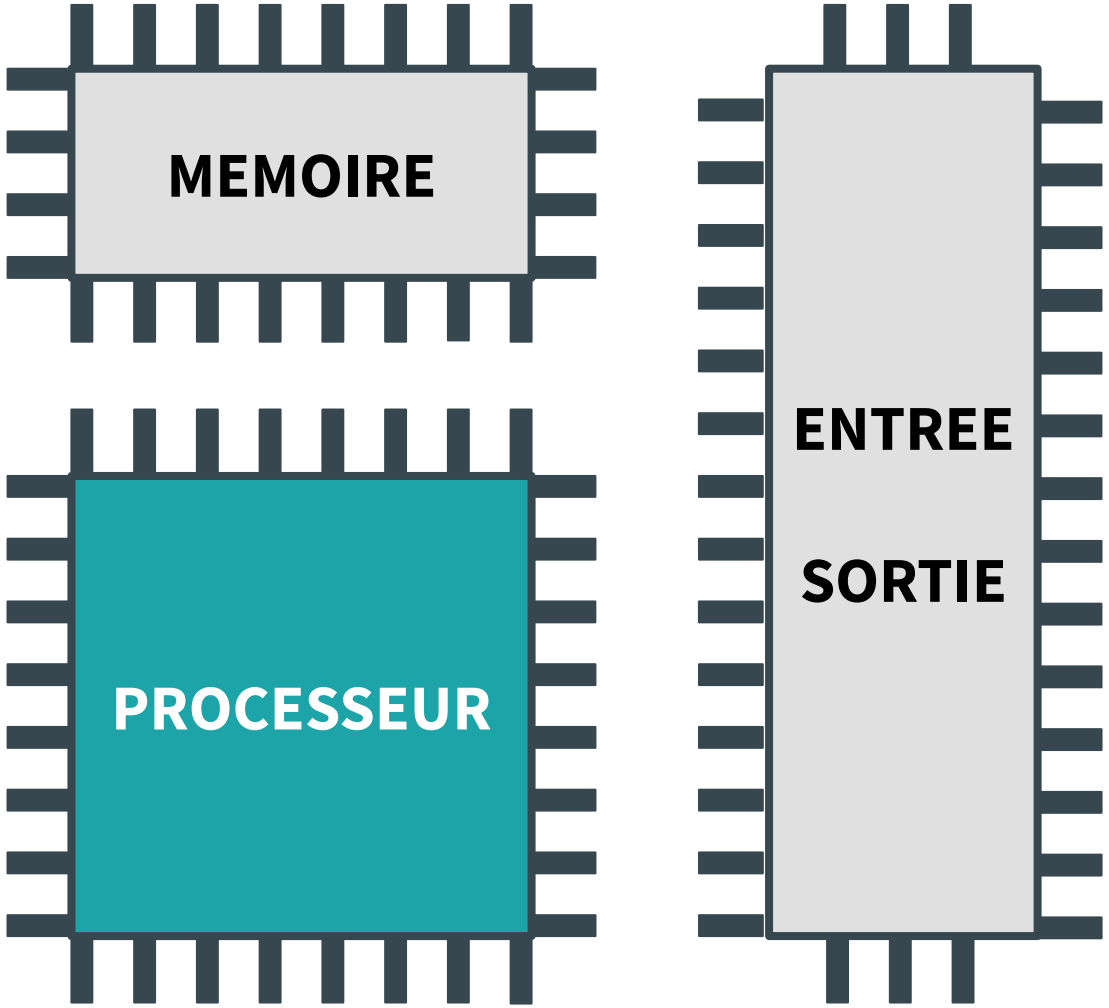

MEMOIRE
PROCESSEUR
ENTREE
SORTIE

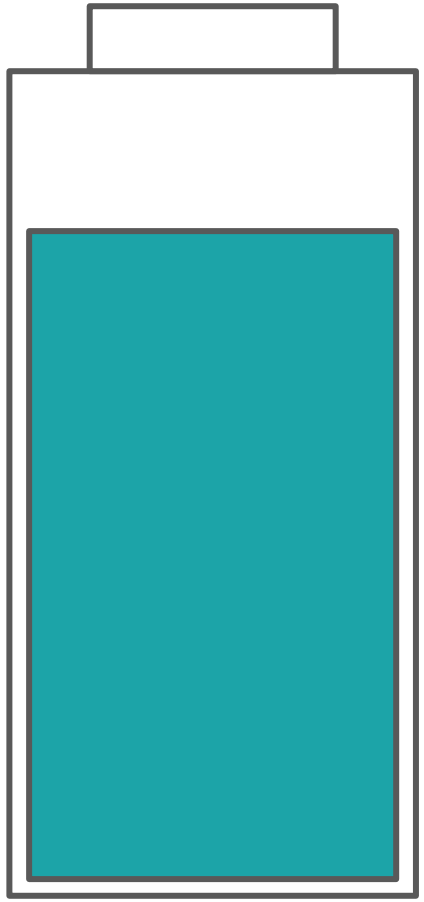

τ_7
τ_6
τ_8
τ_5
τ_1
τ_4
τ_2
τ_3

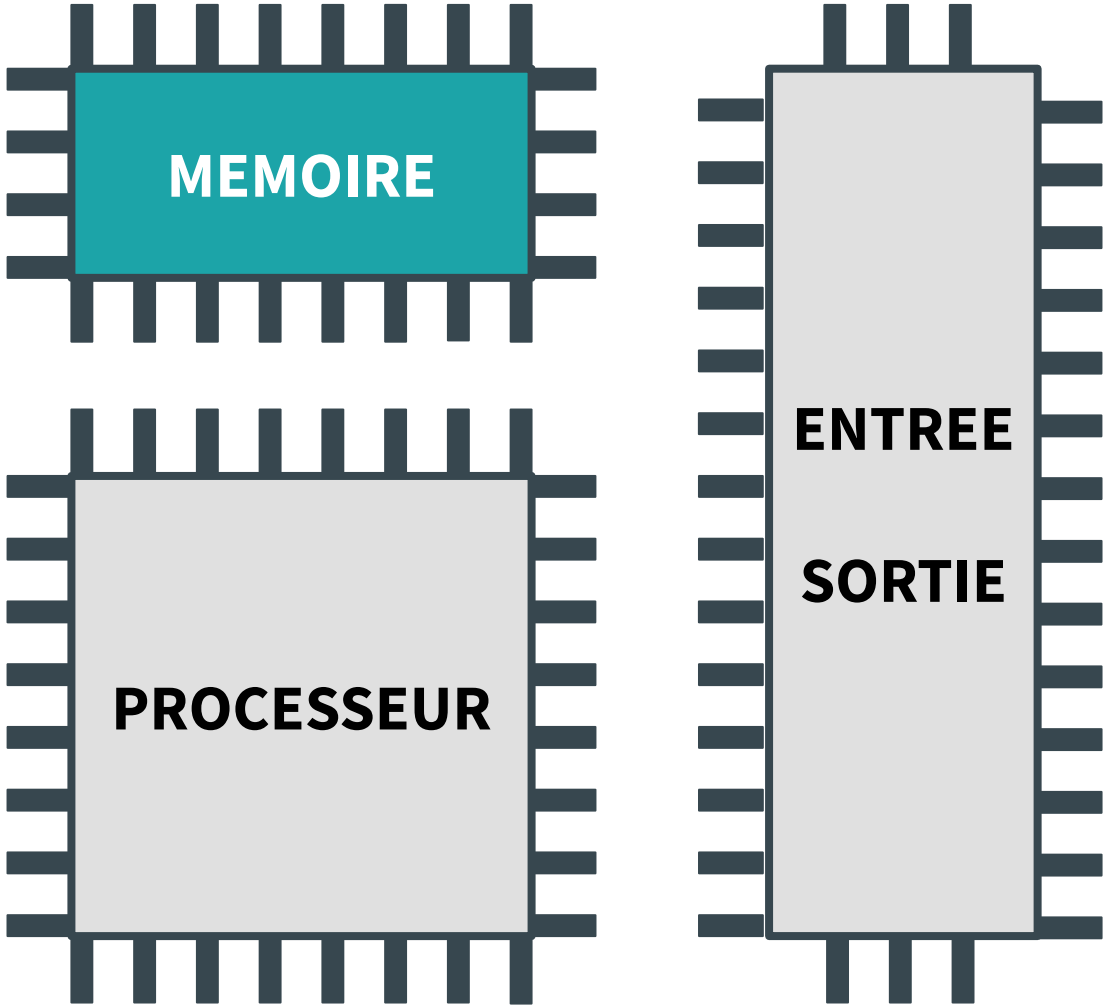

MEMOIRE
PROCESSEUR
ENTREE
SORTIE
τ_7
τ_6
τ_8
τ_5
τ_1
τ_4
τ_2
τ_3

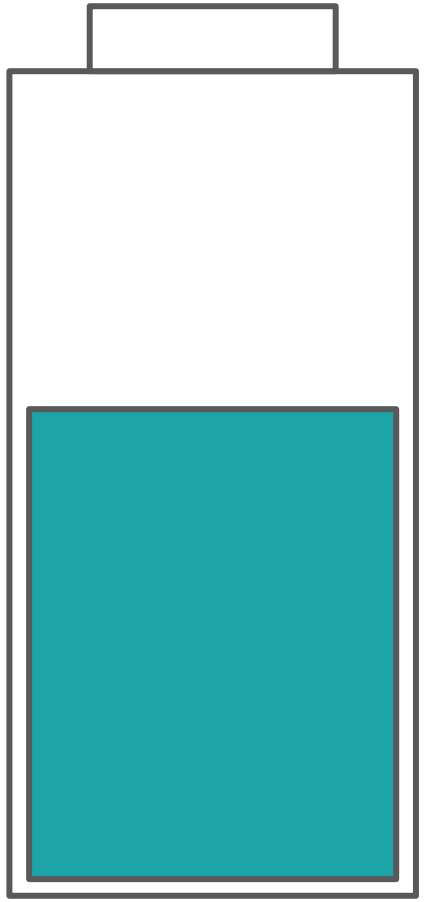

MEMOIRE
PROCESSEUR
ENTREE
SORTIE

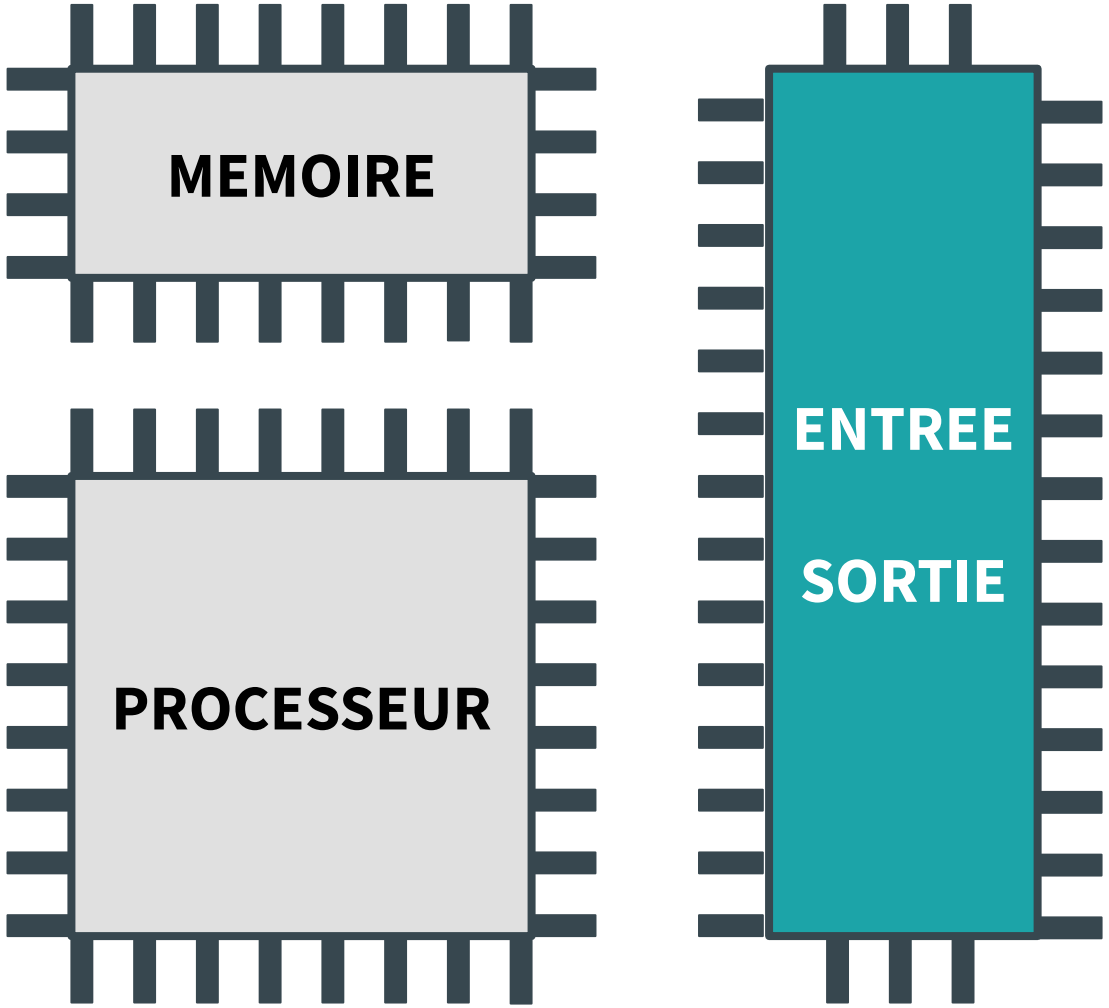


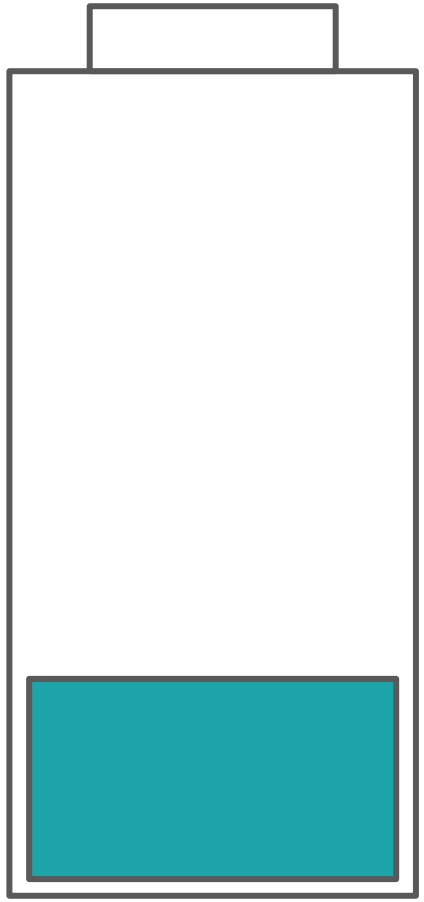

MEMOIRE
PROCESSEUR
ENTREE
SORTIE

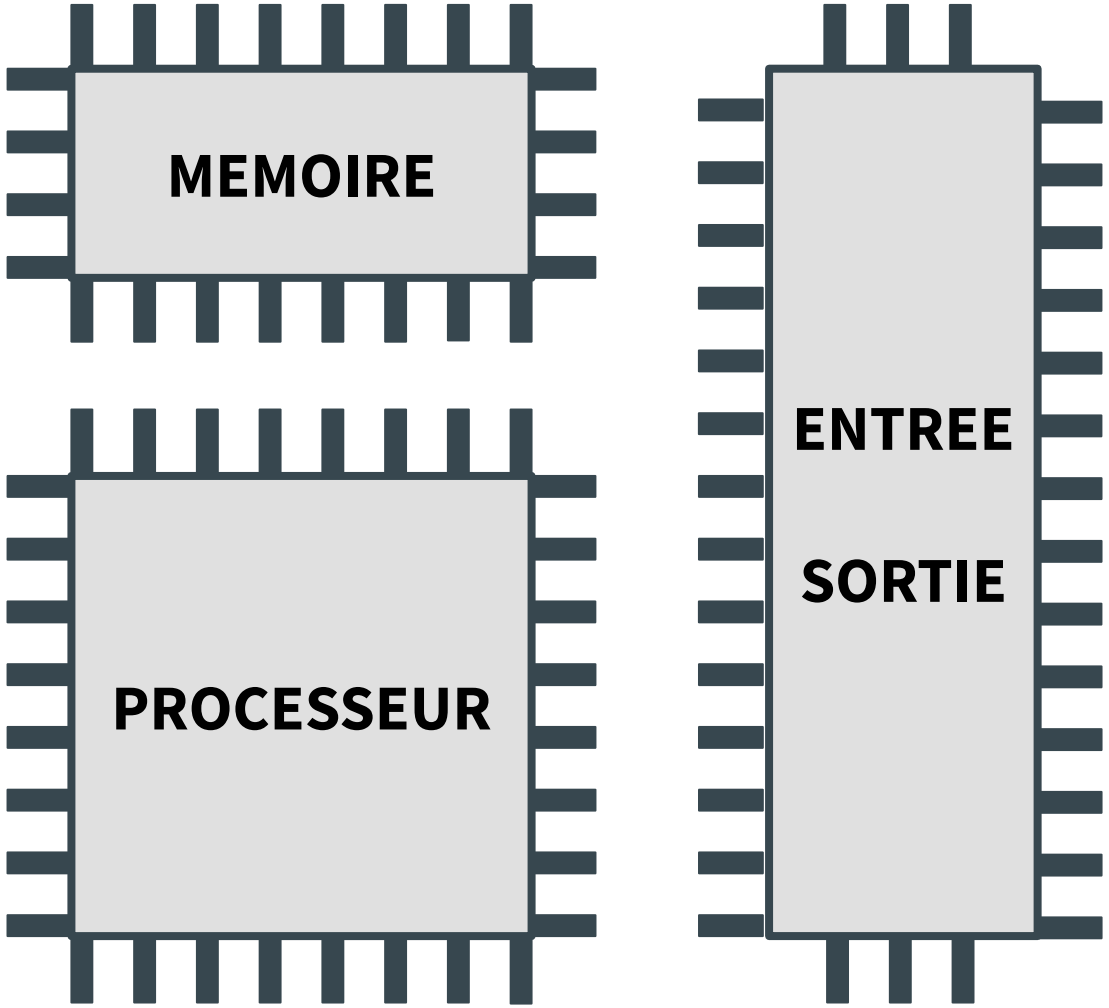


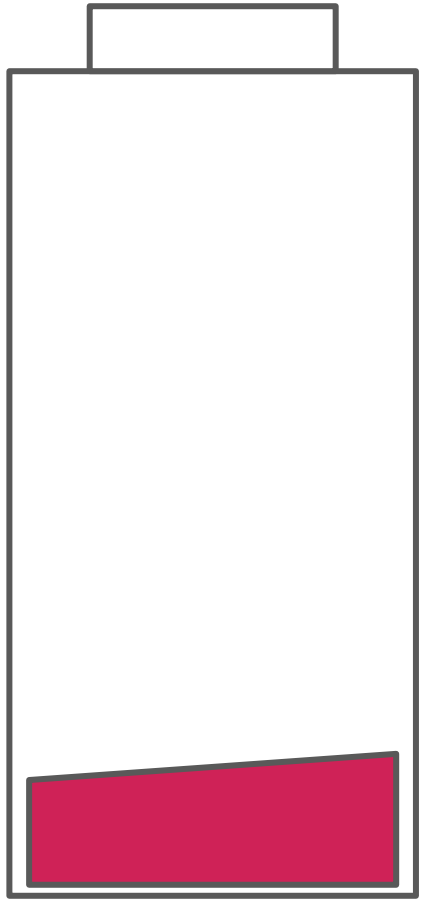

MEMOIRE
PROCESSEUR
ENTREE
SORTIE

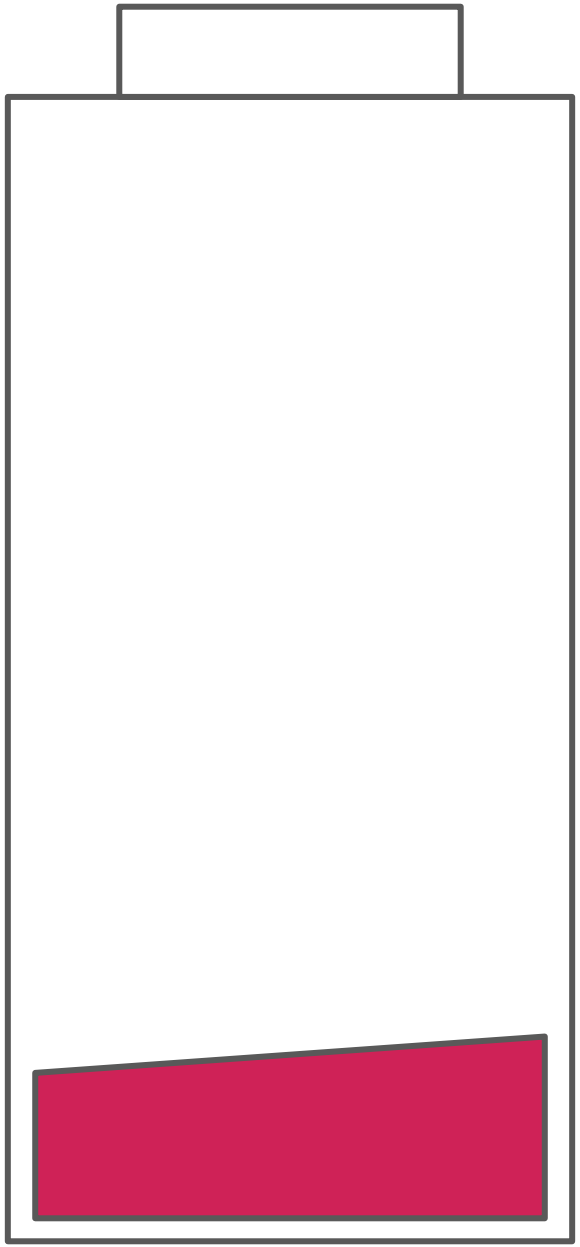


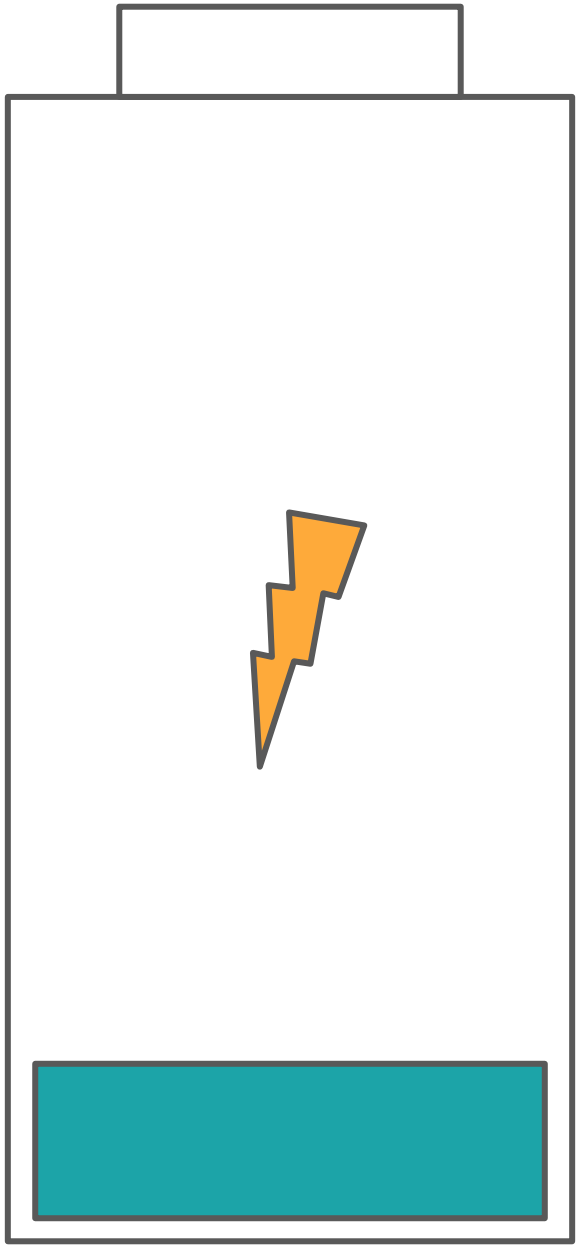

MEMOIRE
PROCESSEUR
ENTREE
SORTIE

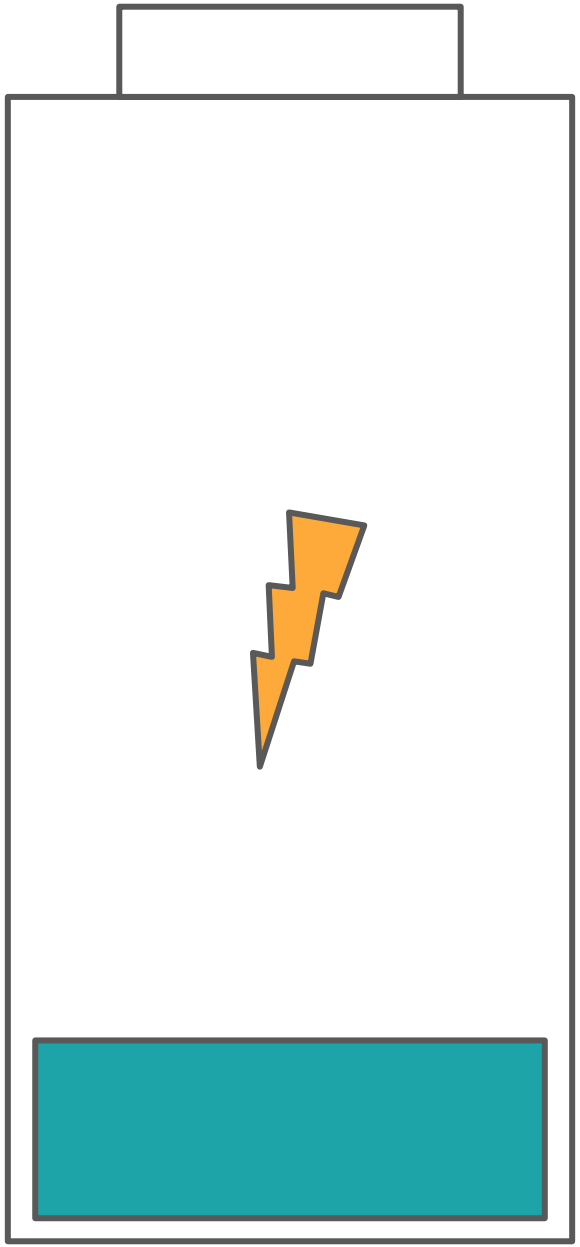


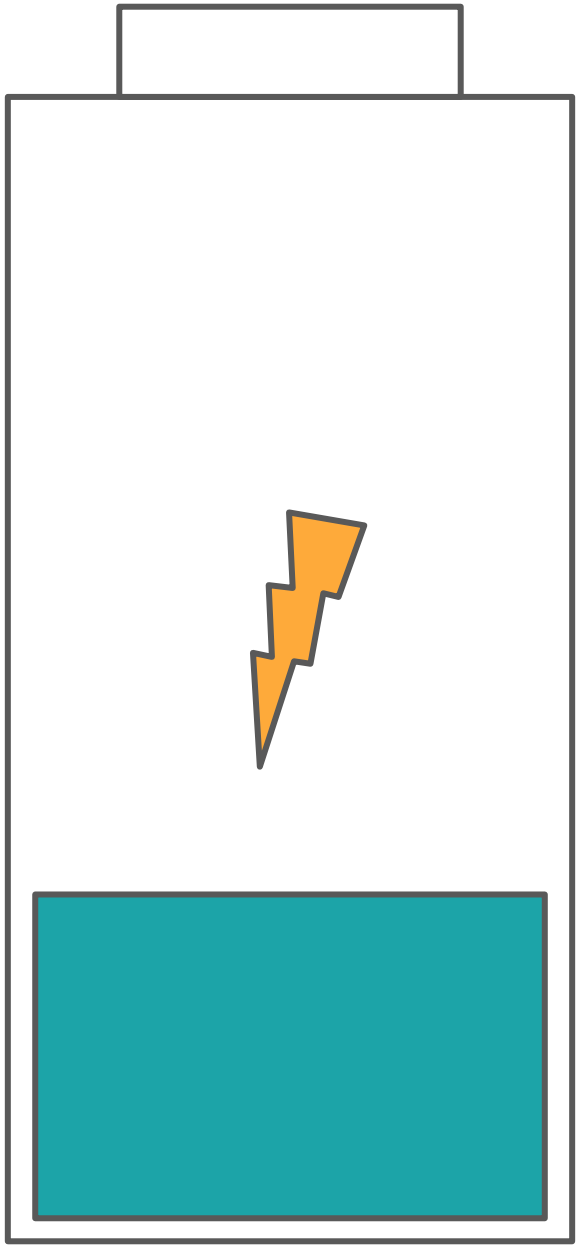

MEMOIRE
PROCESSEUR
ENTREE
SORTIE

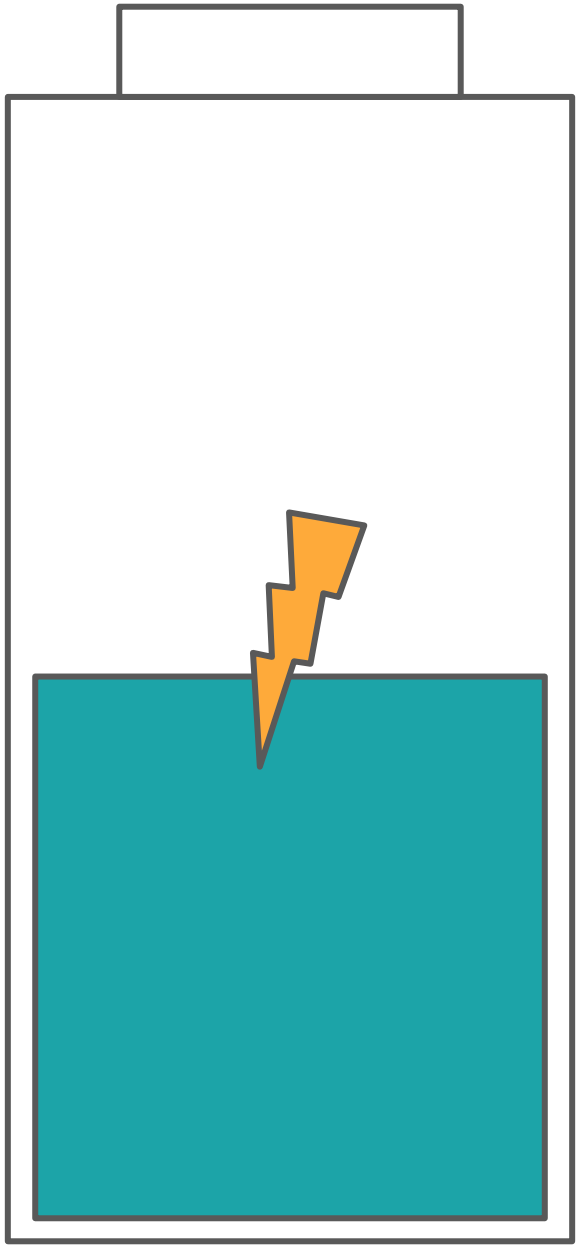


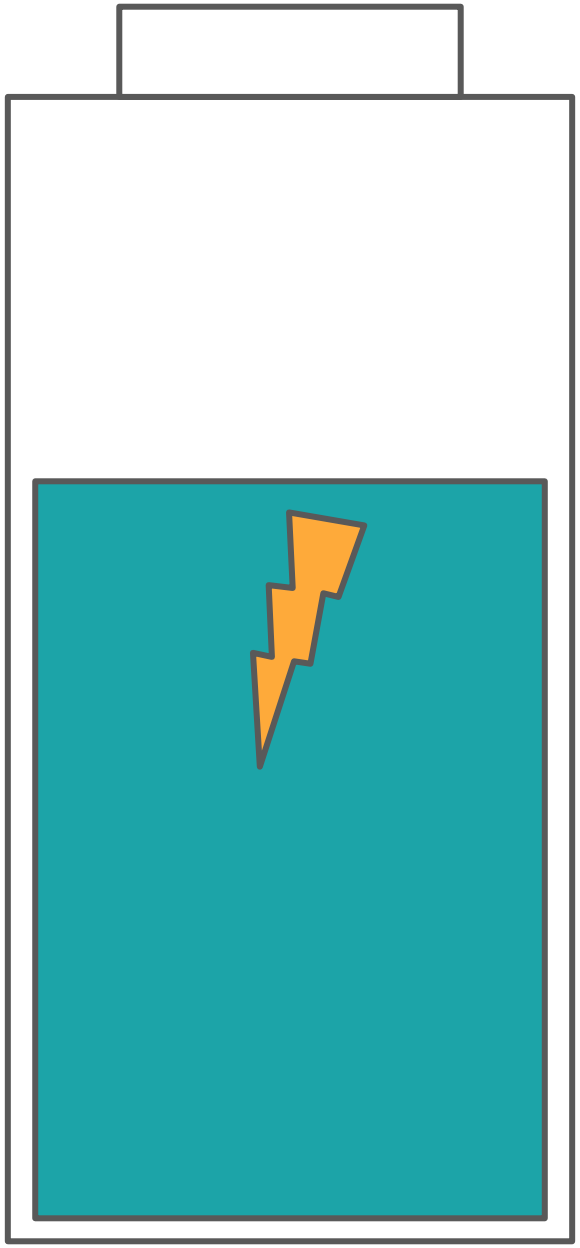


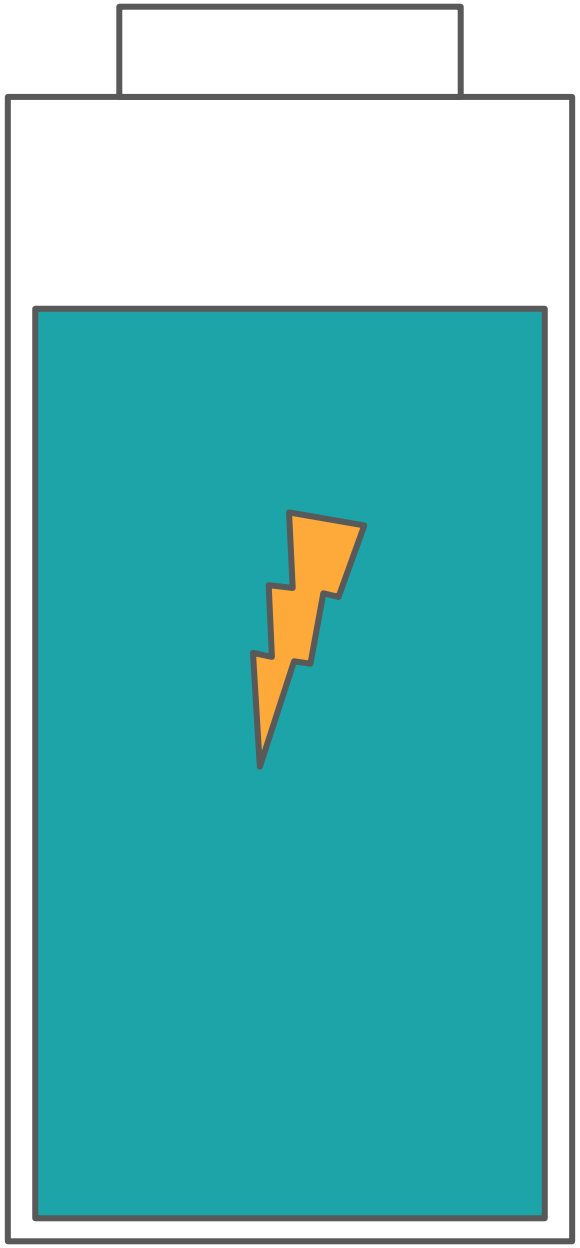


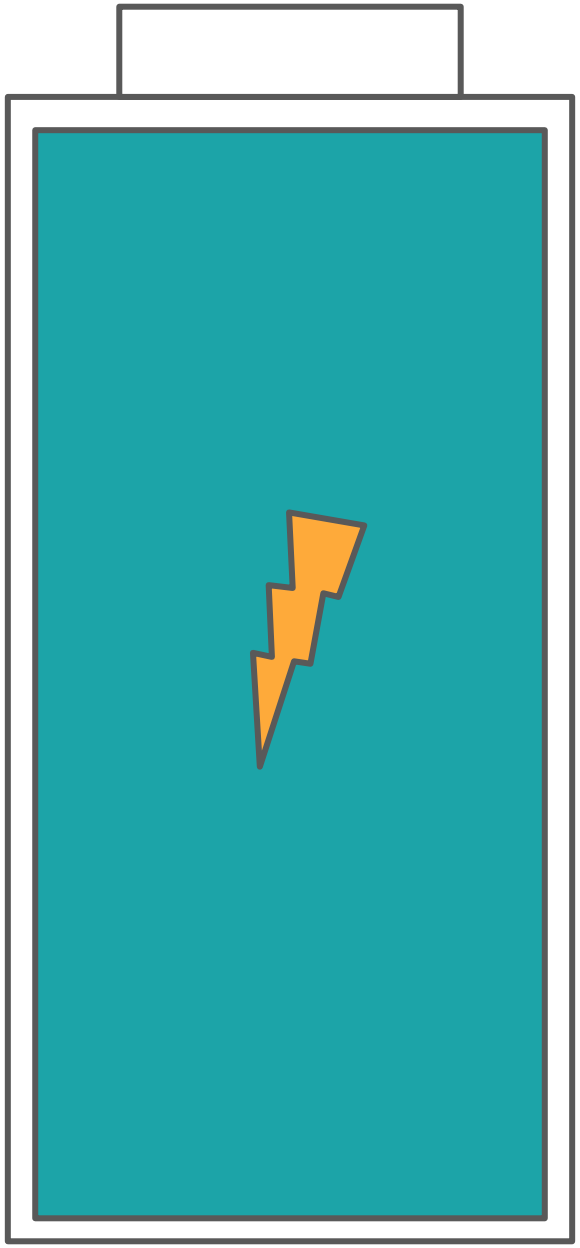


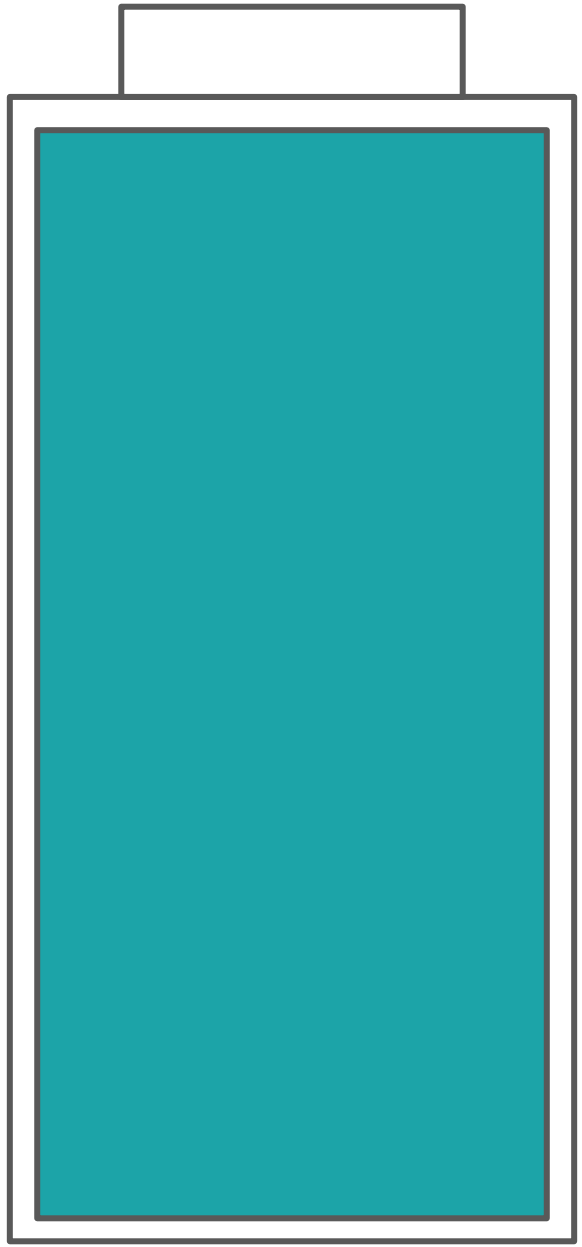


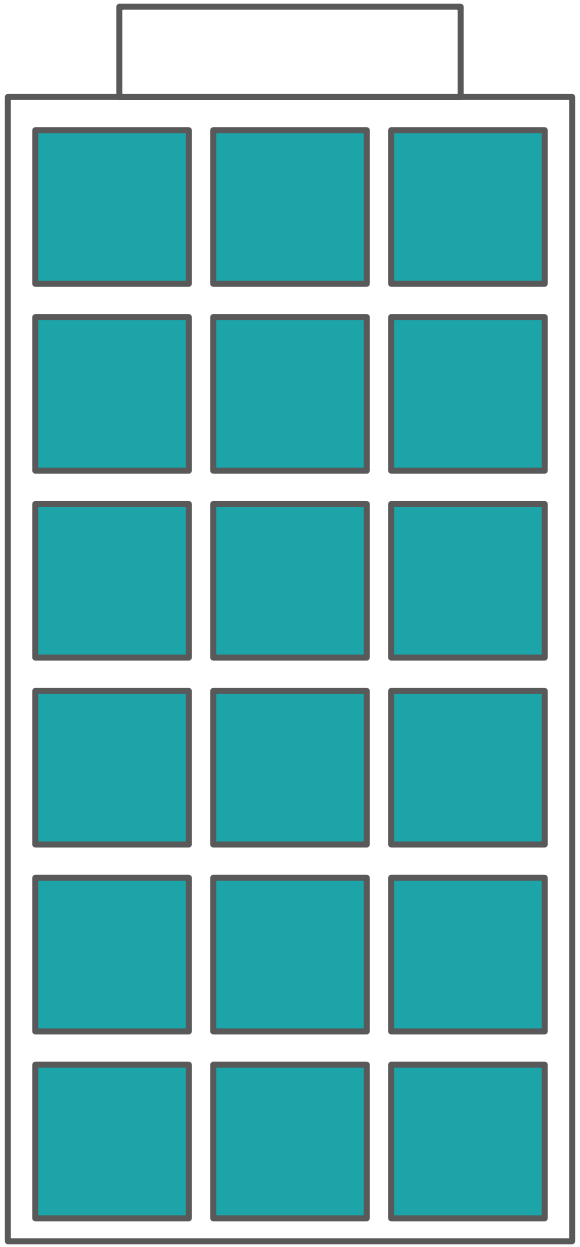


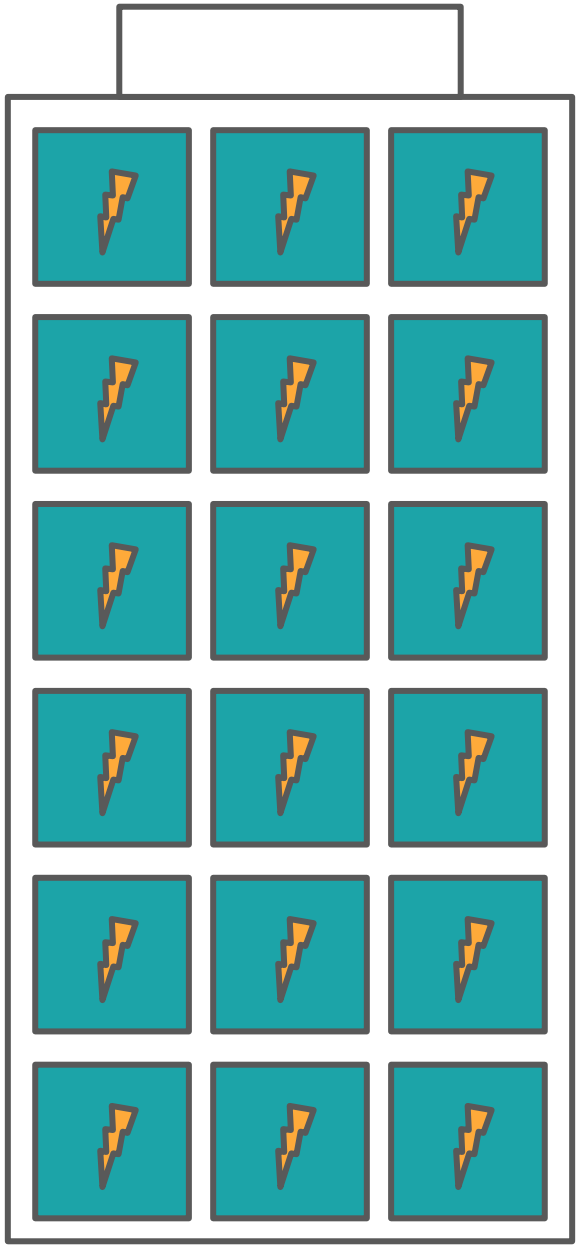


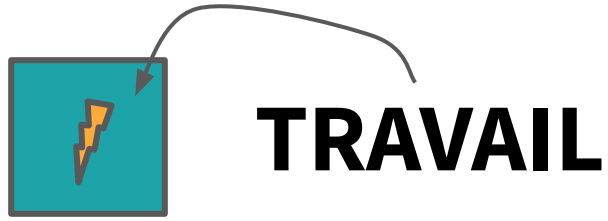




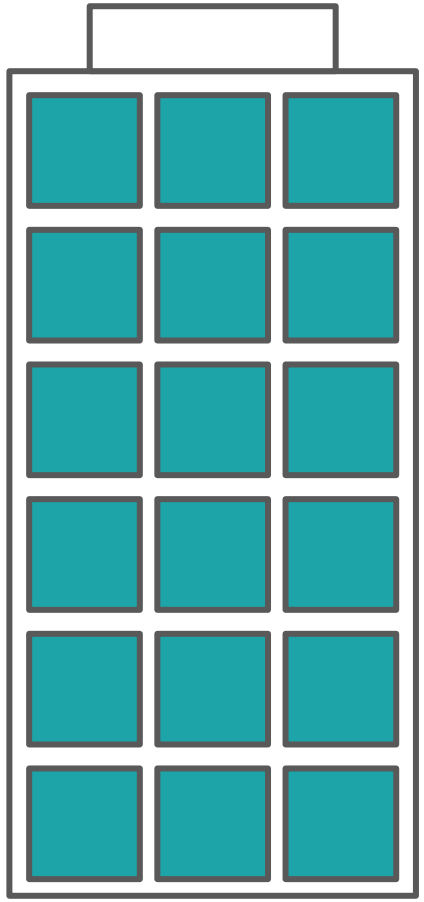


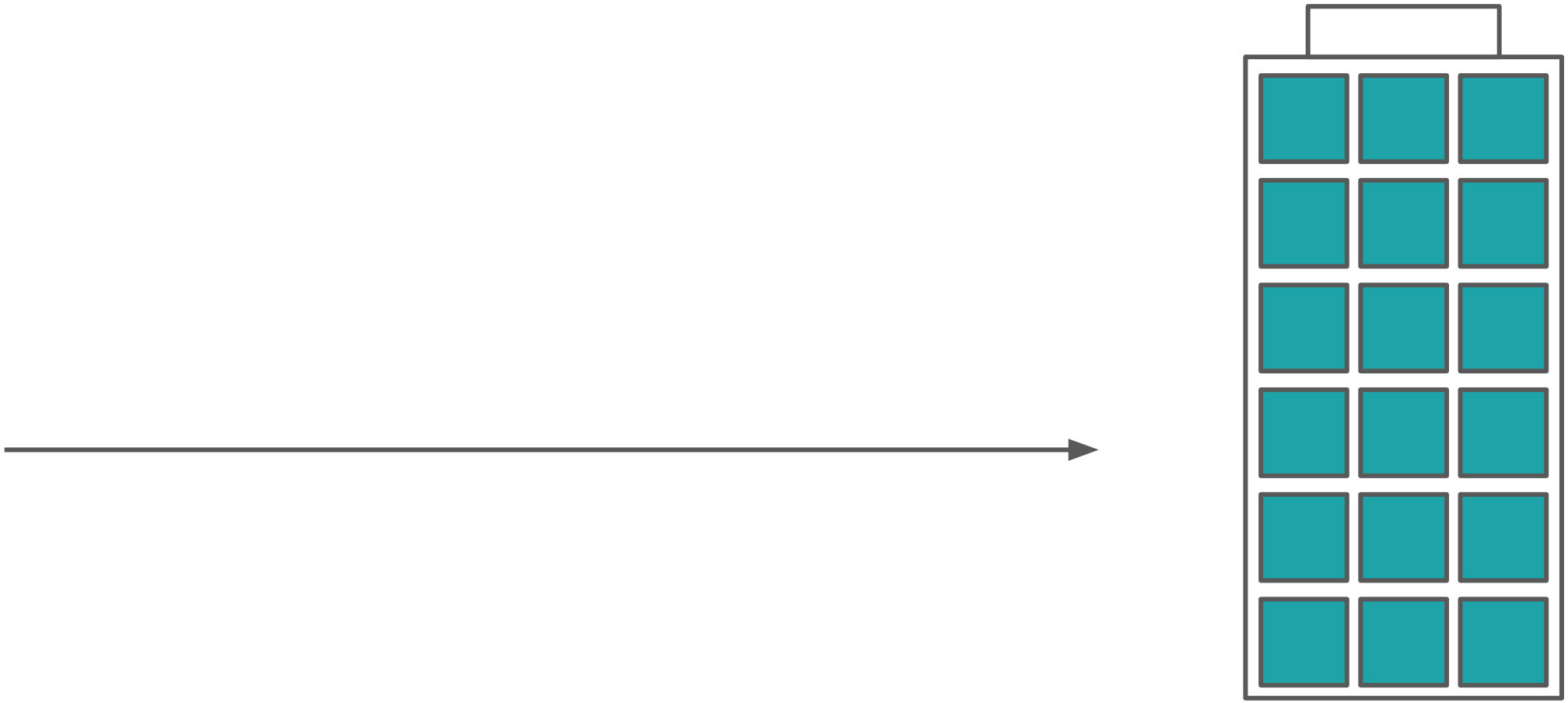


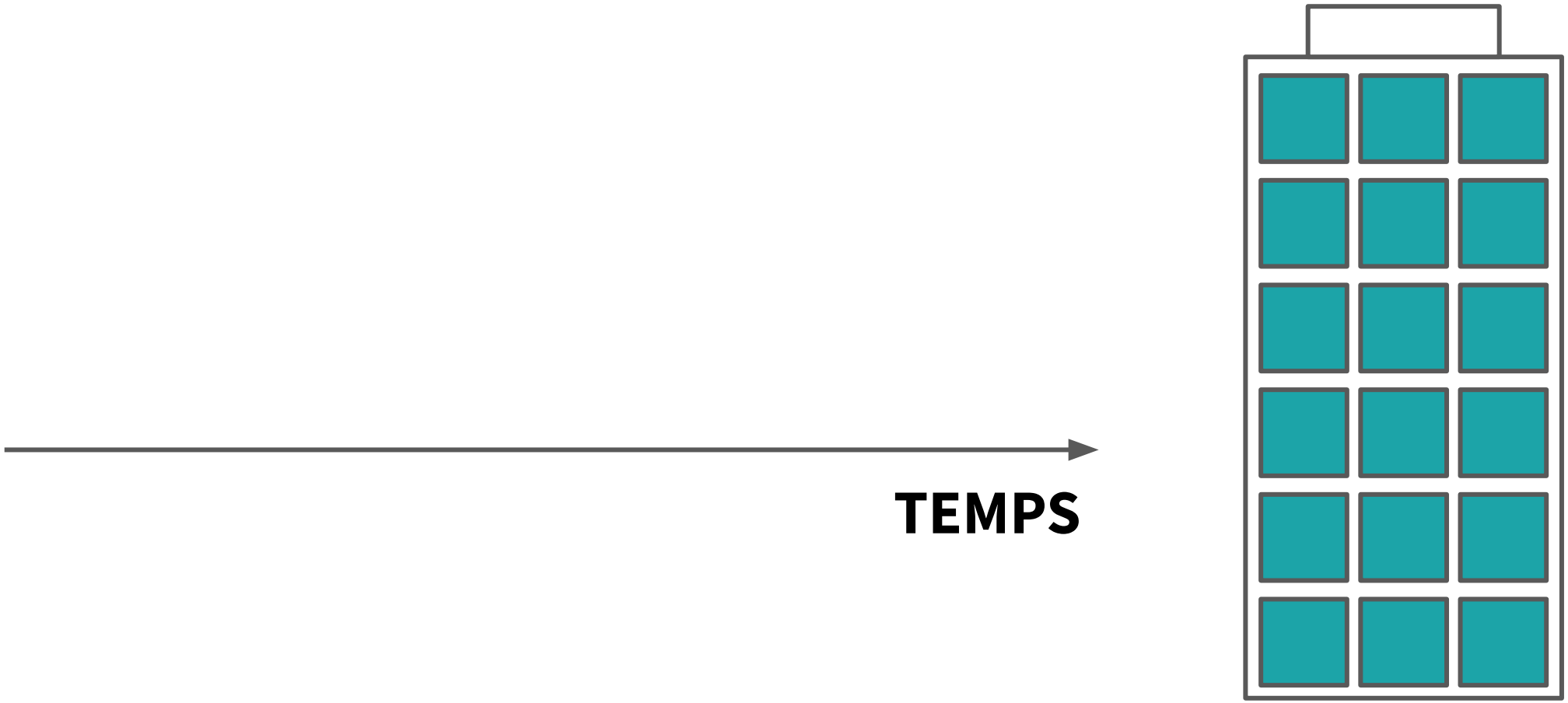


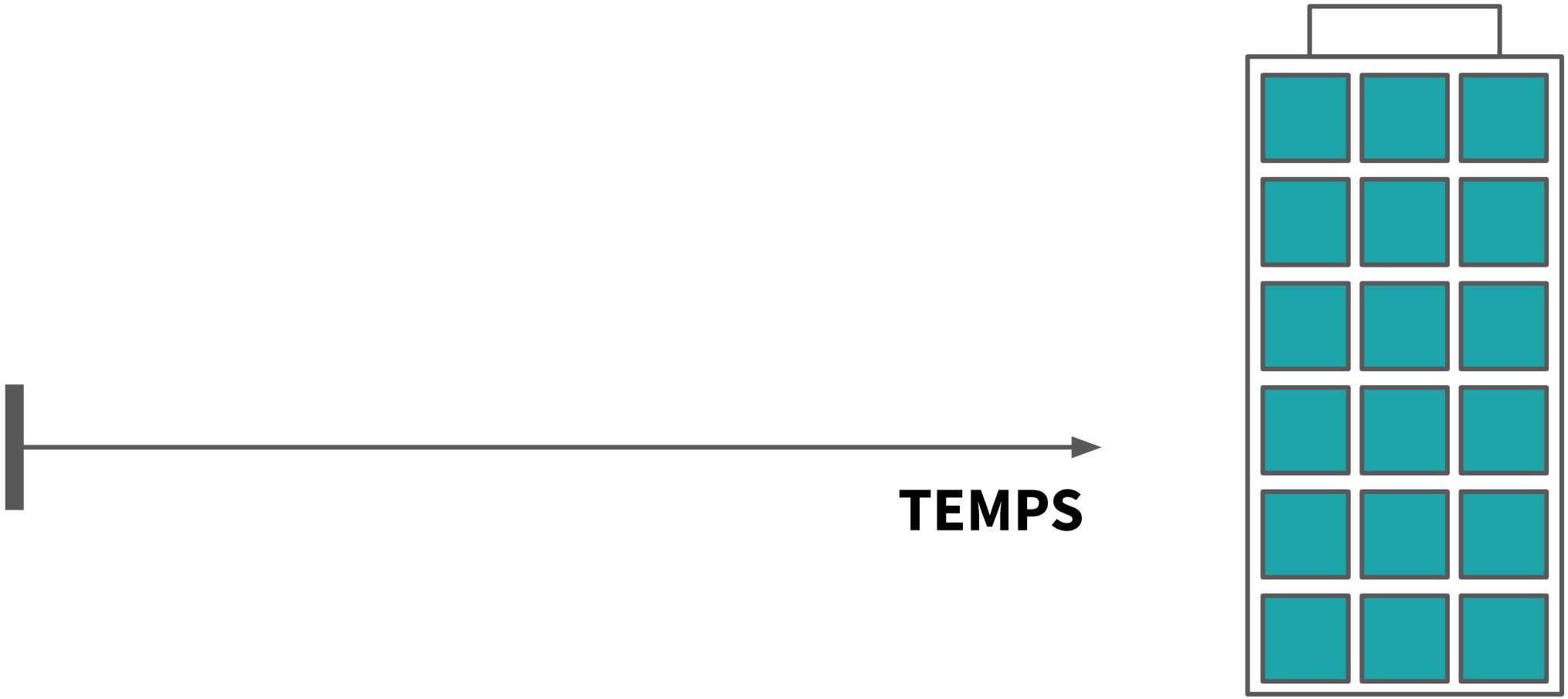

TRAVAIL

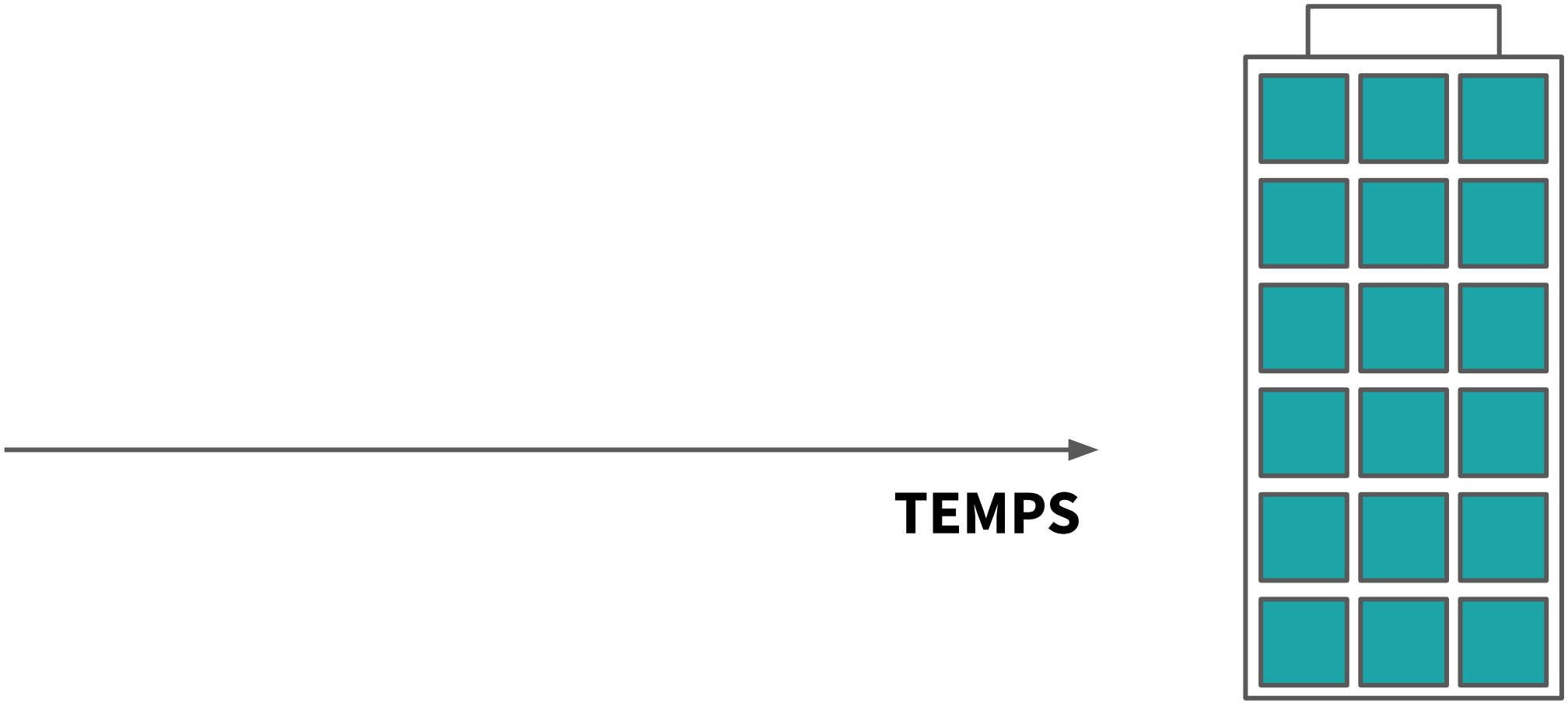
TEMPS

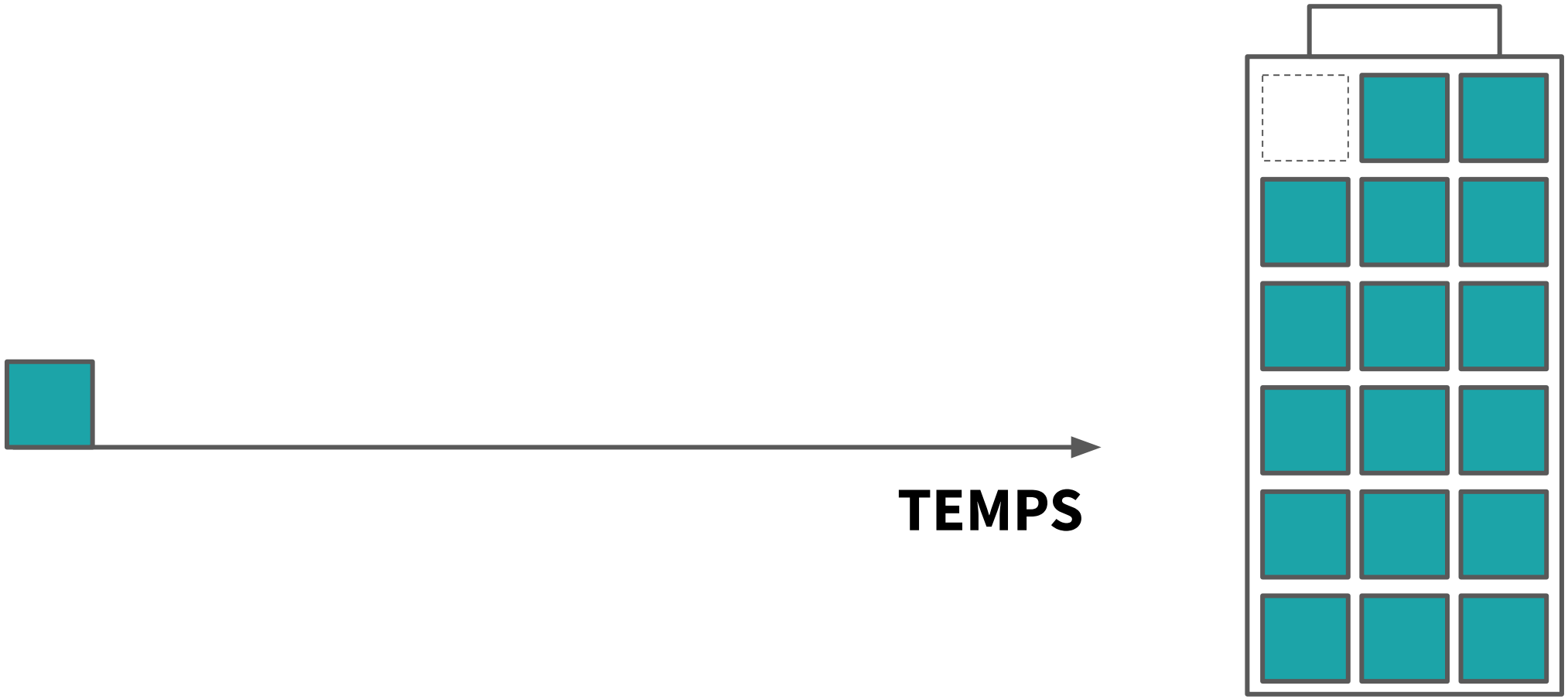


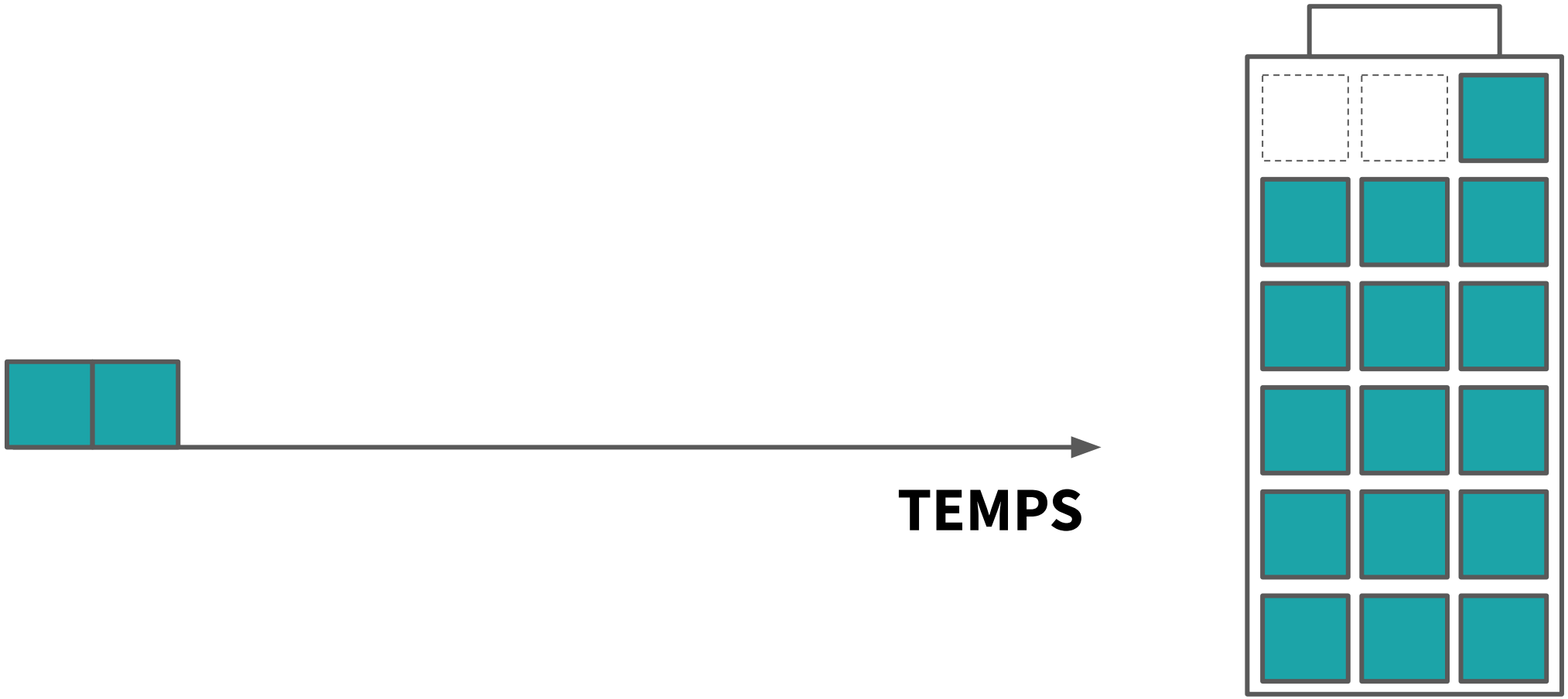



TEMPS

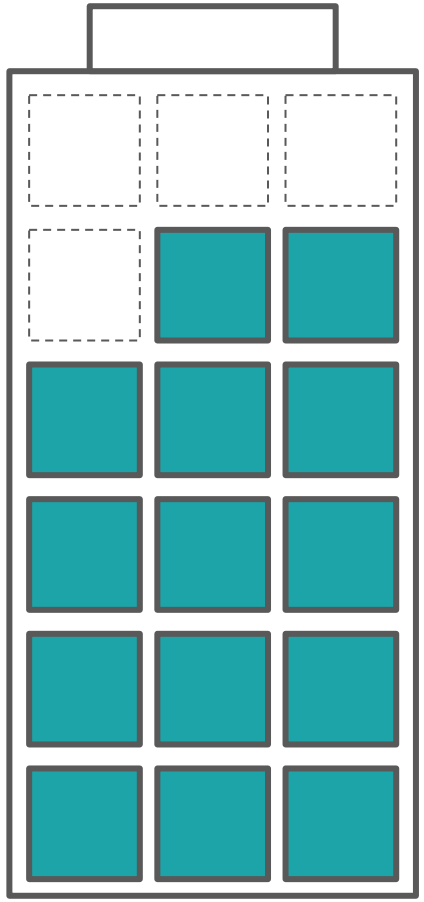

TEMPS


TEMPS

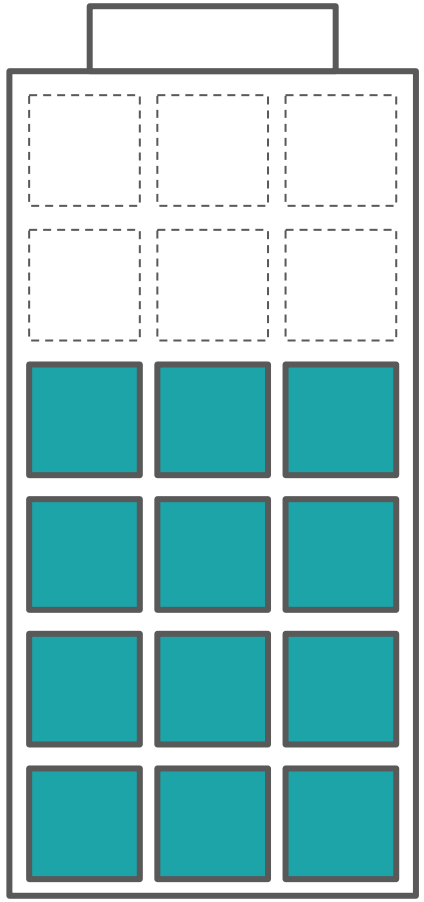

TEMPS


TEMPS

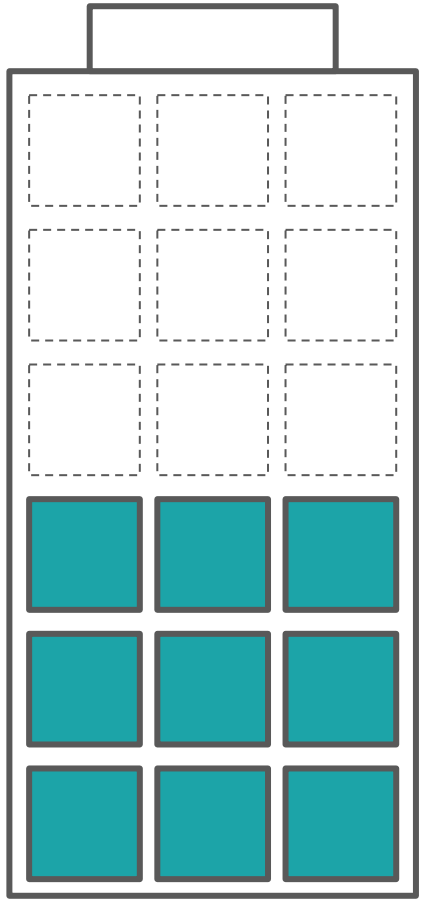
TEMPS

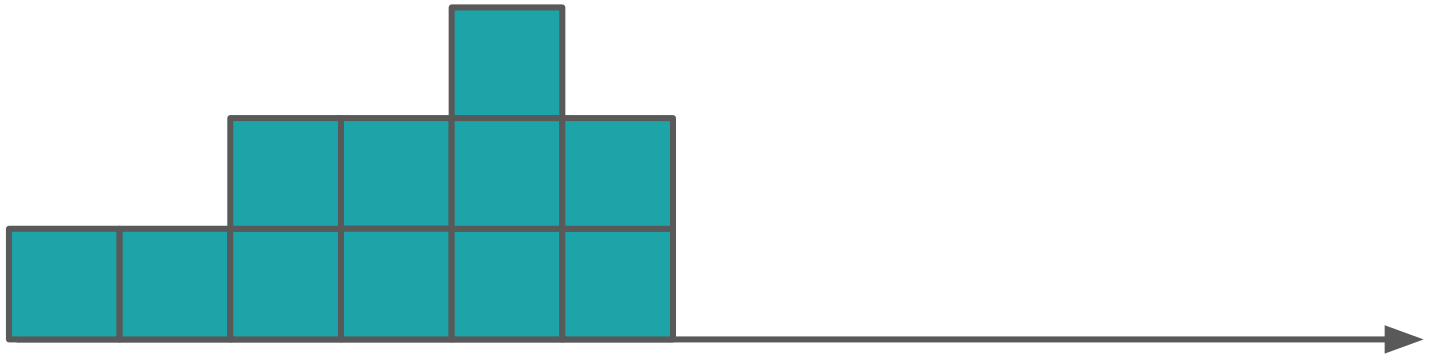


TEMPS

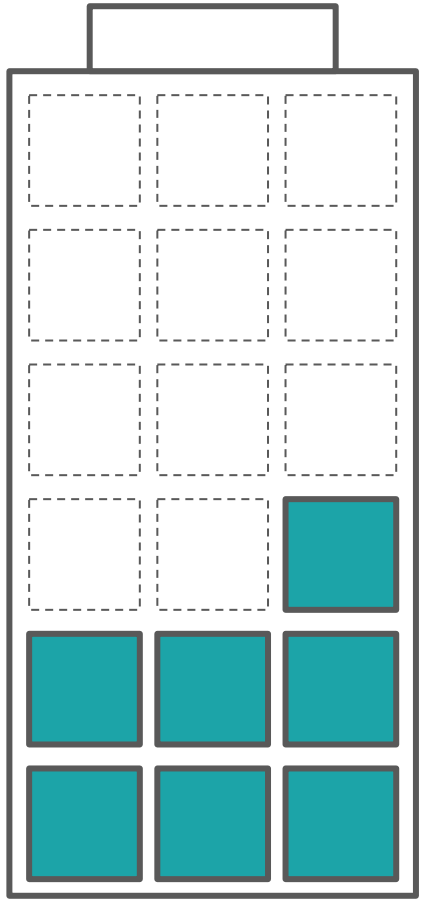


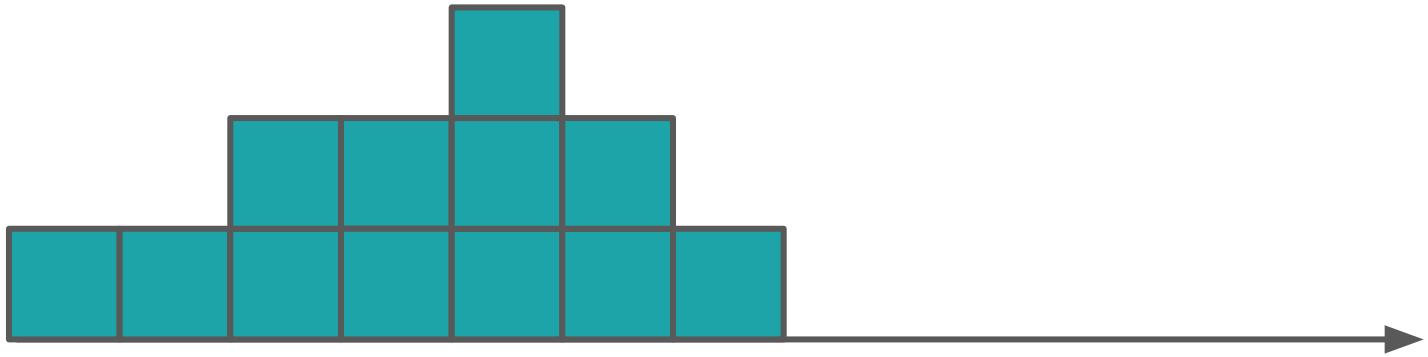
TEMPS



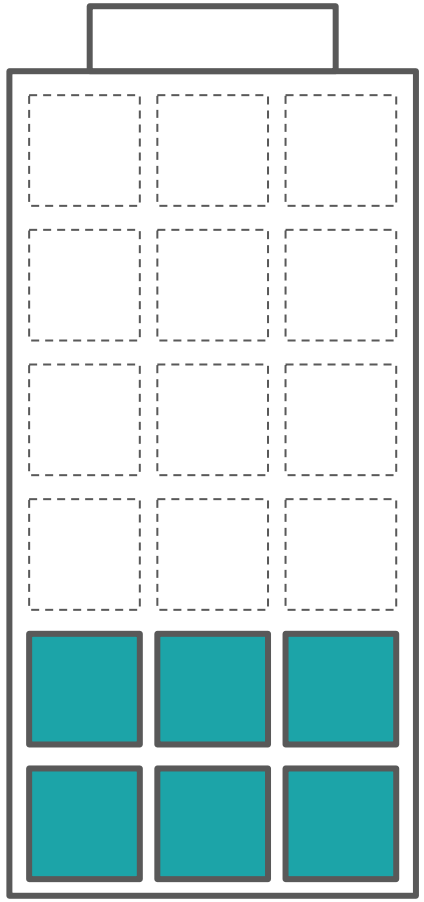


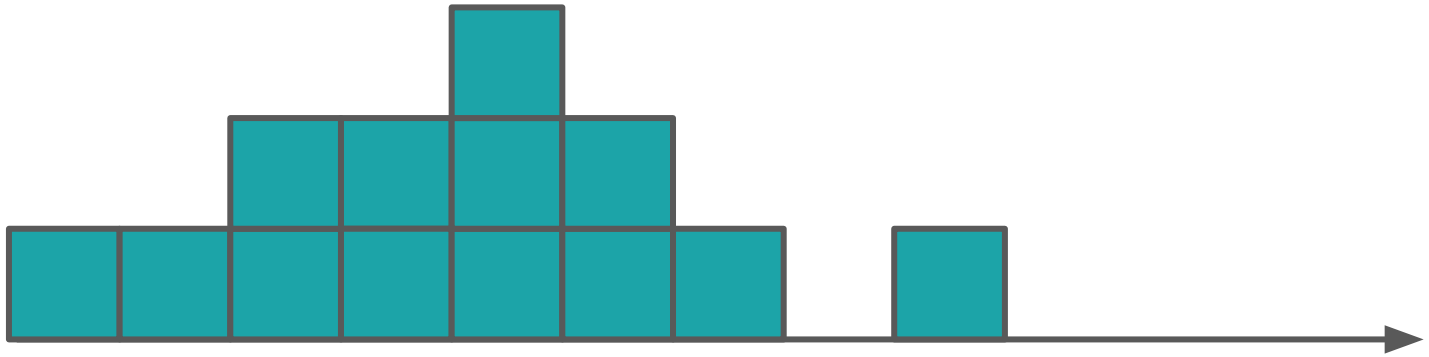
TEMPS



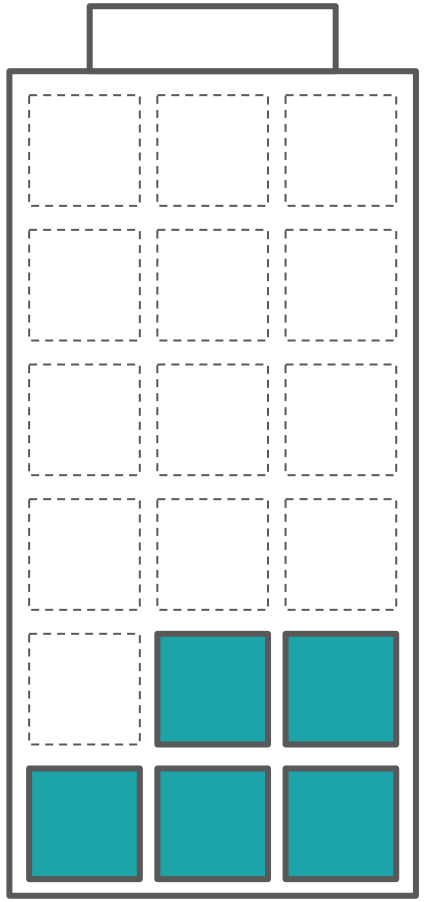


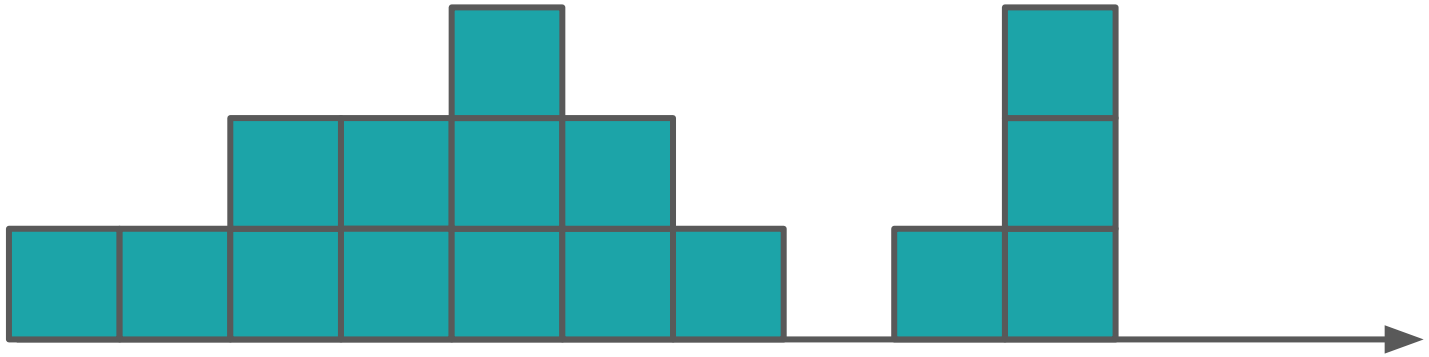
TEMPS



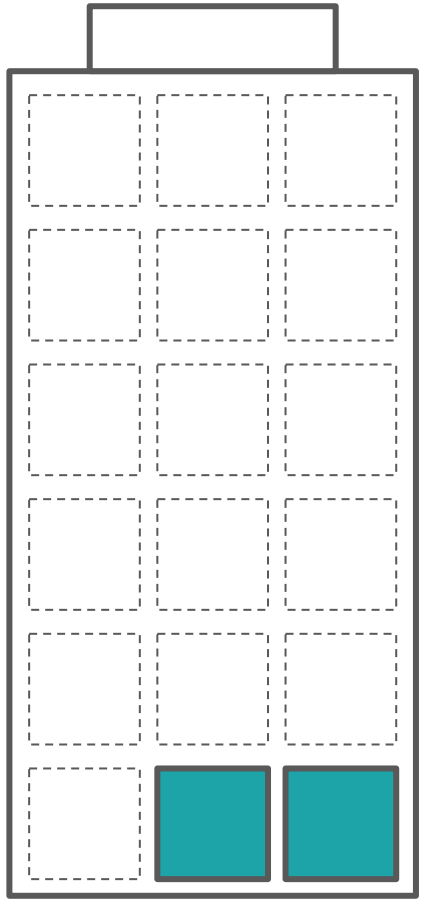


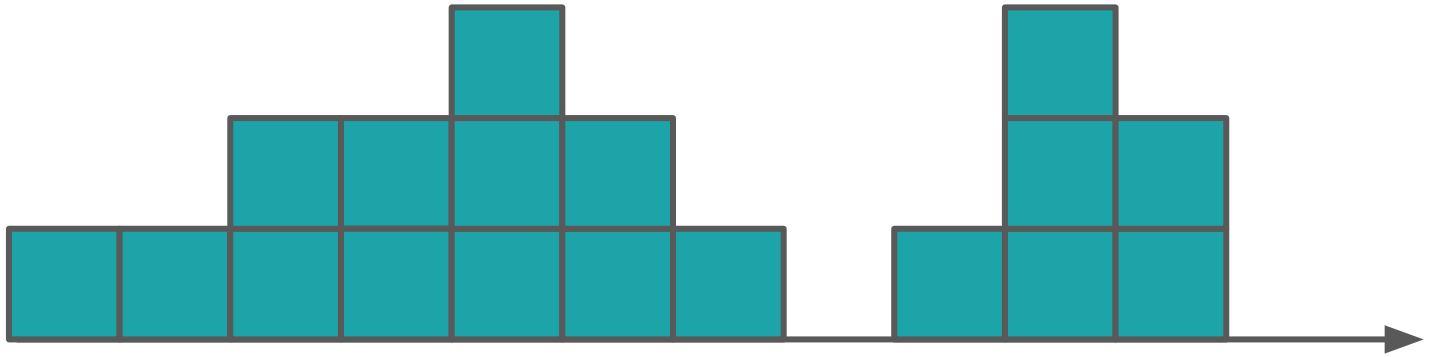
TEMPS



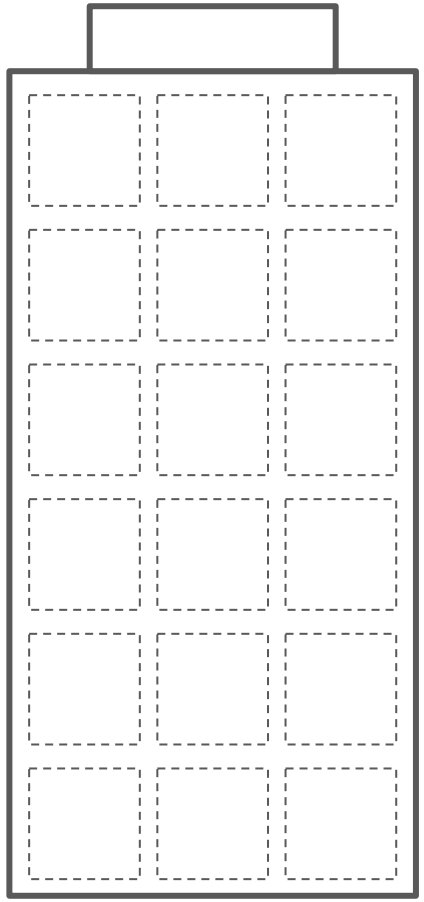


TEMPS





TEMPS



TRAVAIL ?

TRAVAIL ?

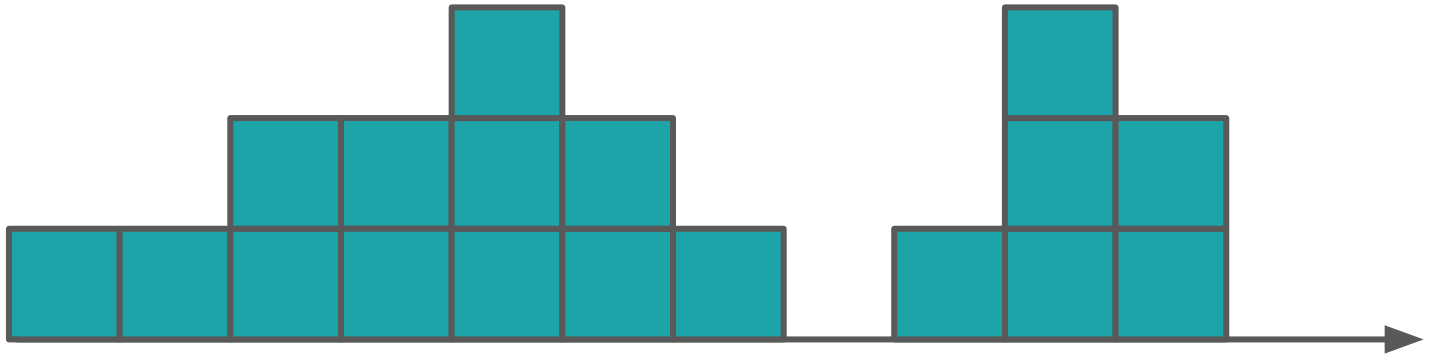
→ ECHEANCES

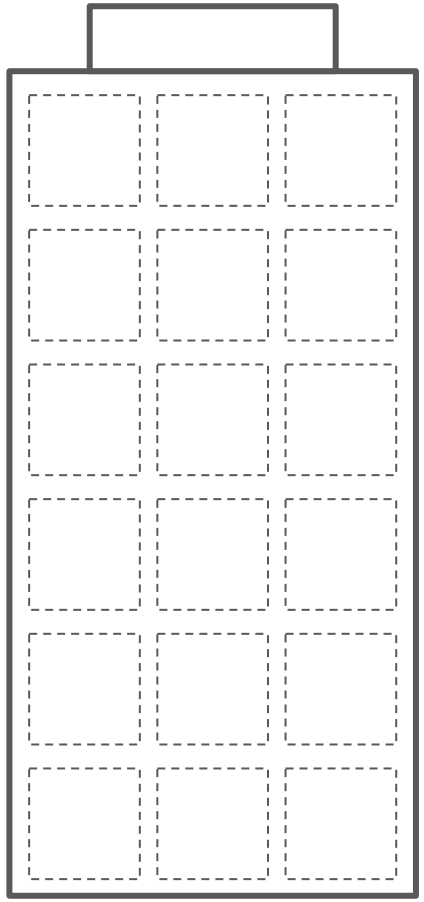
TRAVAIL ? 

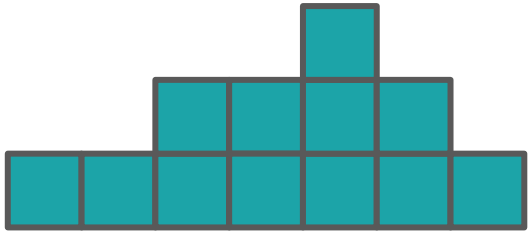
→ ECHEANCES

TRAVAIL ? 

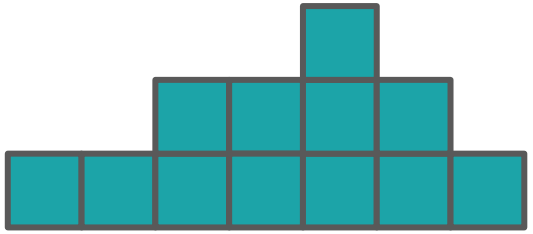
→ ECHEANCES ✕


TEMPS



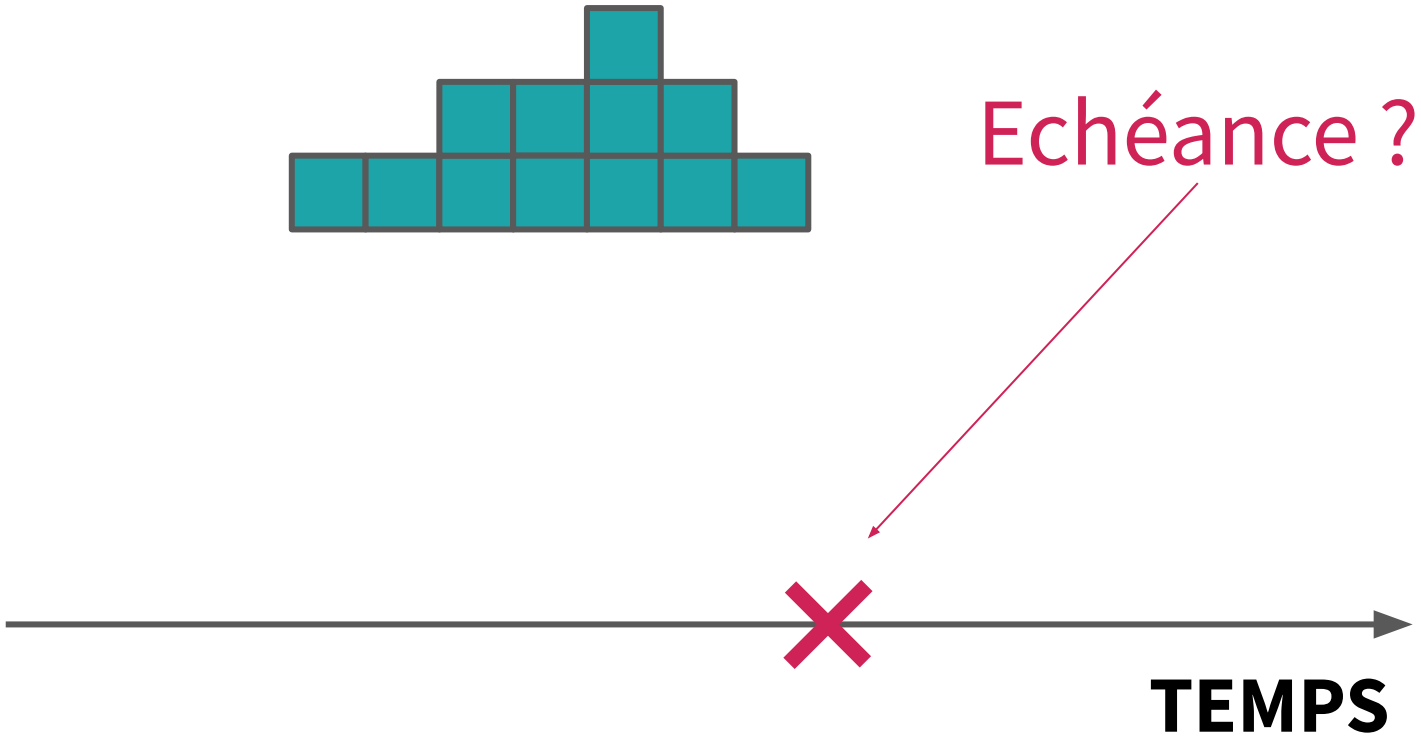


TEMPS

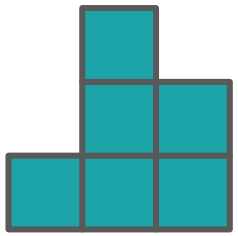


Echéance ?

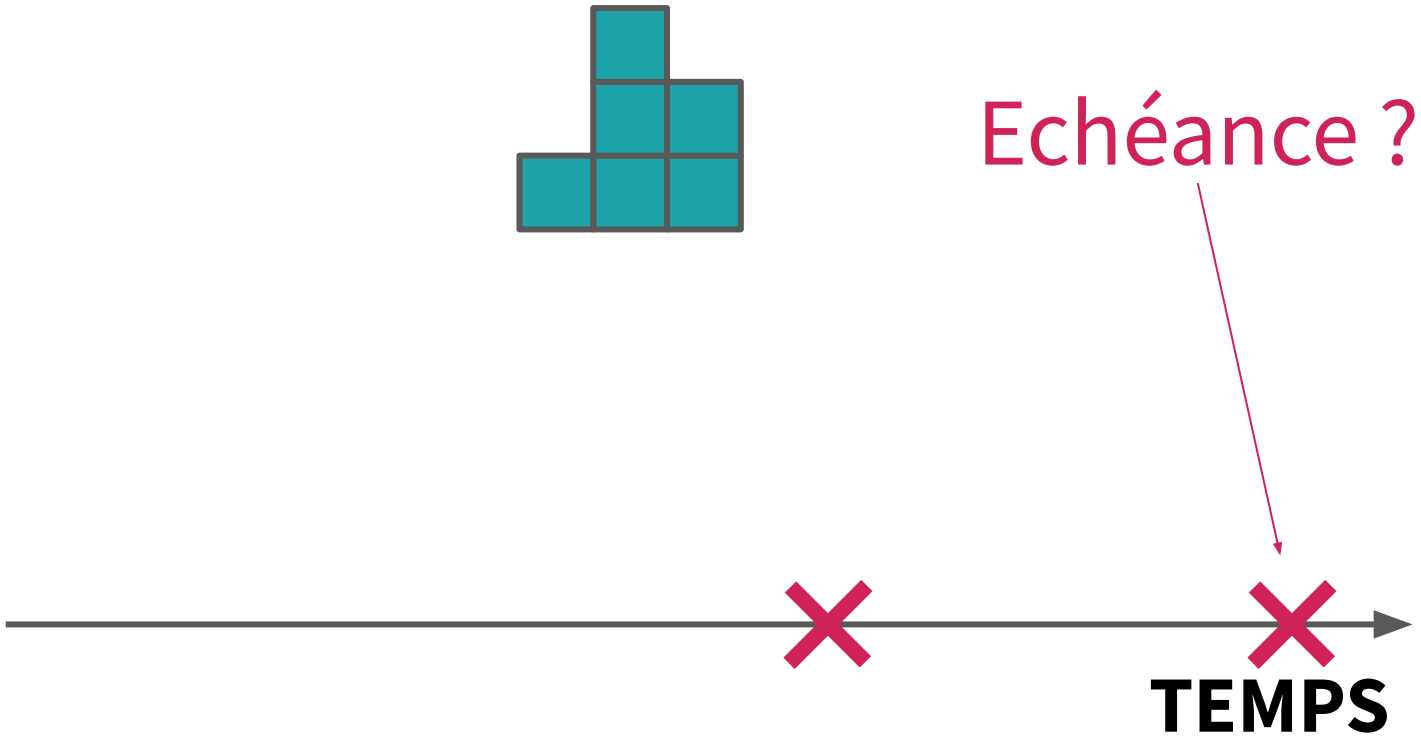
TEMPS


Echéance ?
TEMPS

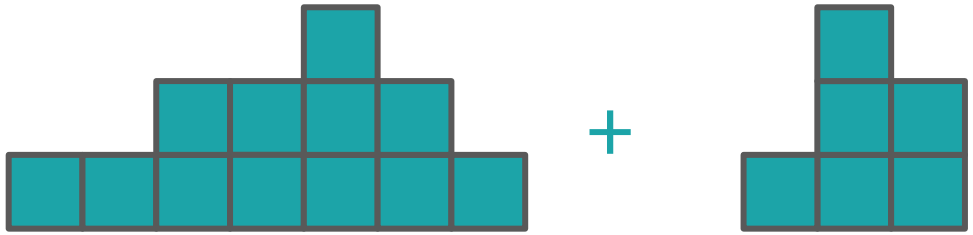

TEMPS



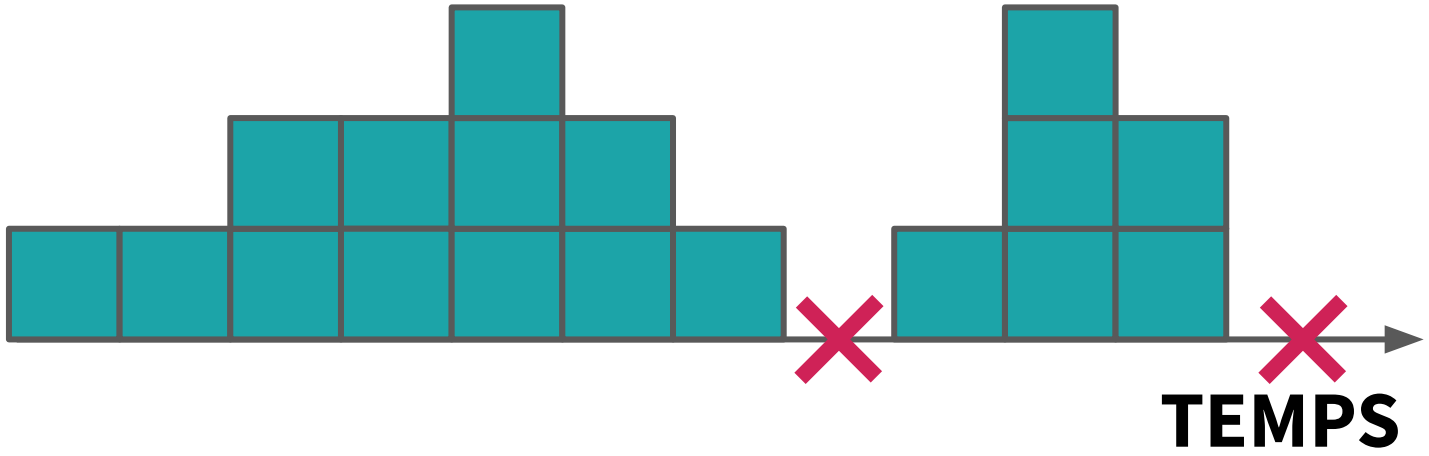
Echéance ?

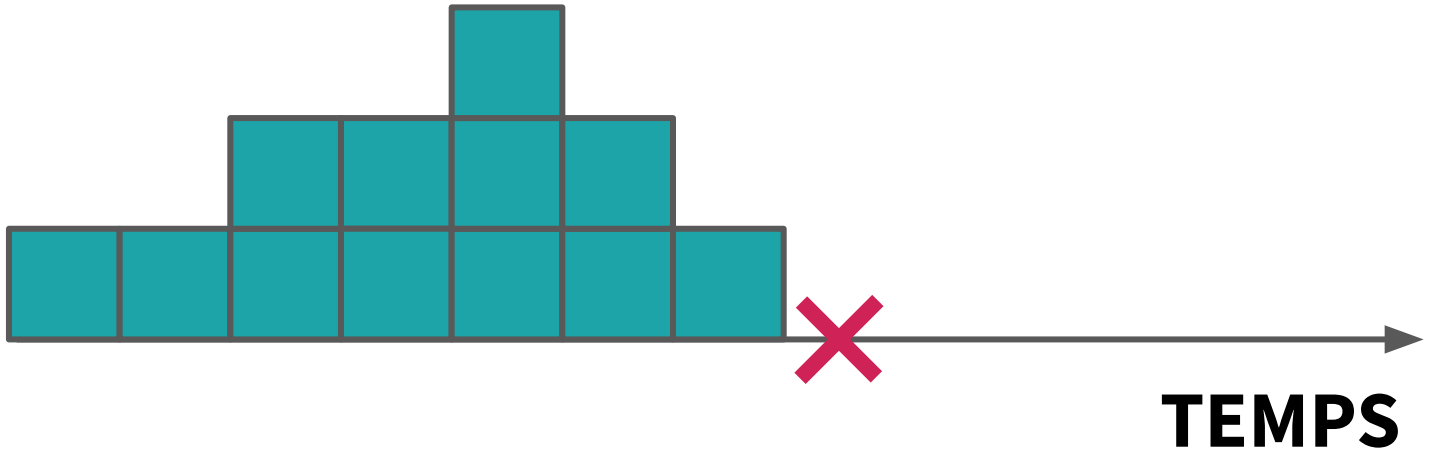

Echéance ?
TEMPS

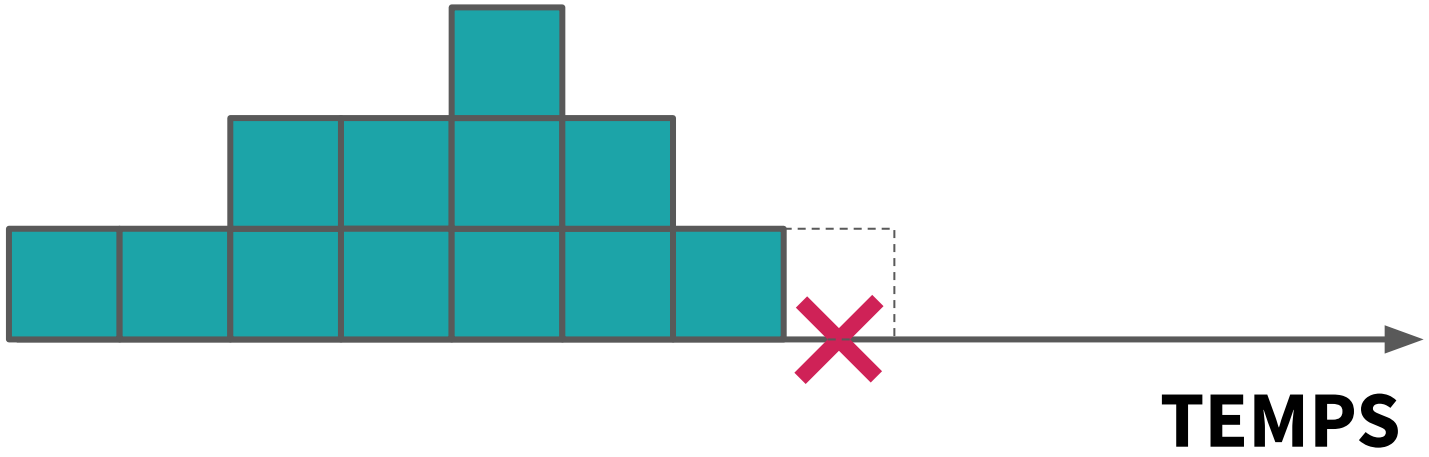

TEMPS

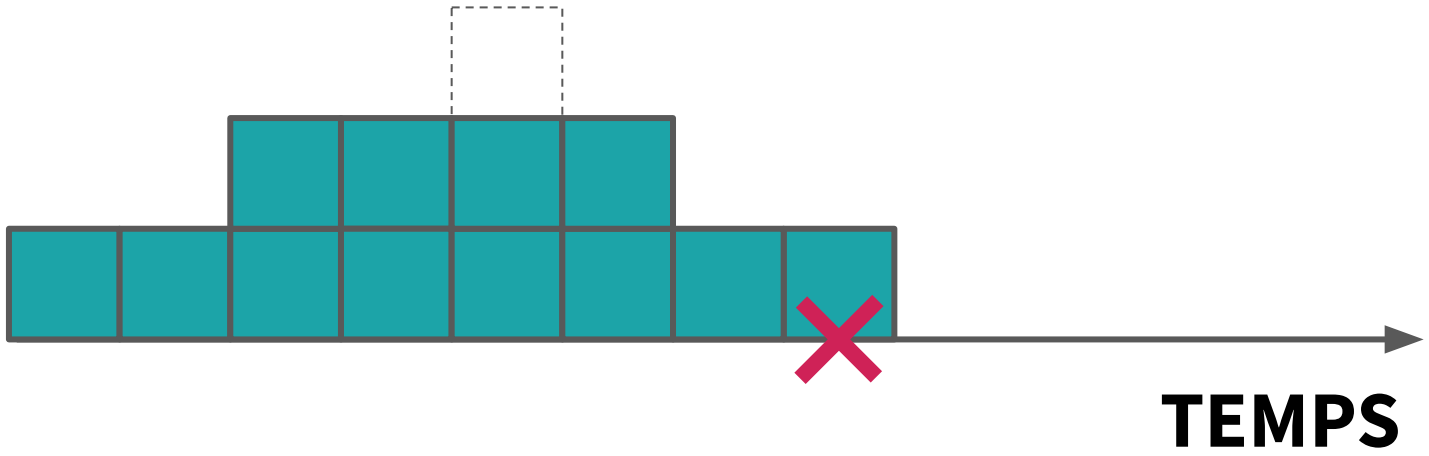



TEMPS

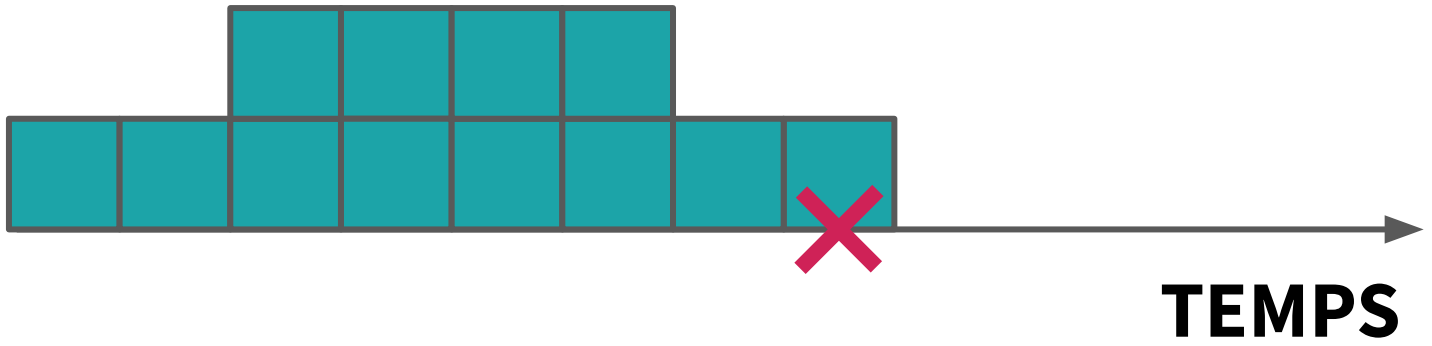

TEMPS


TEMPS

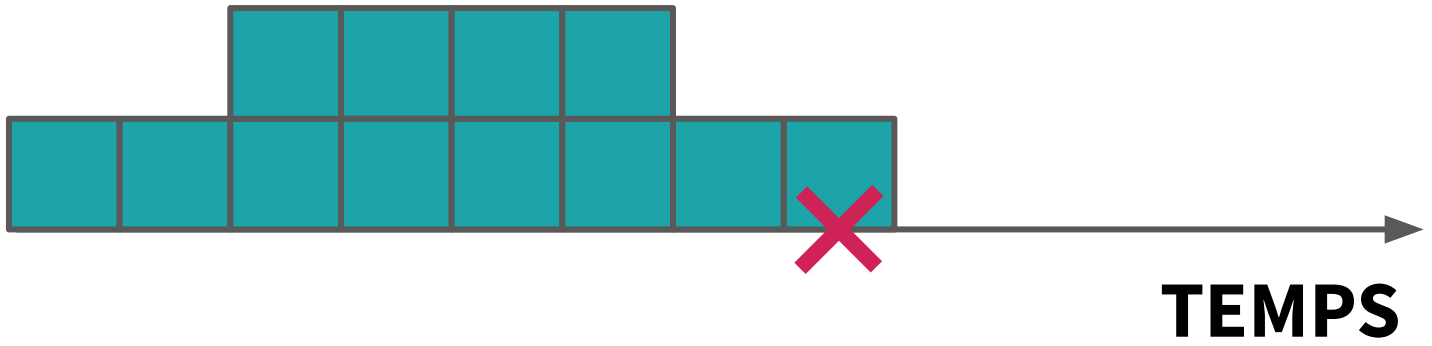

TEMPS


TEMPS

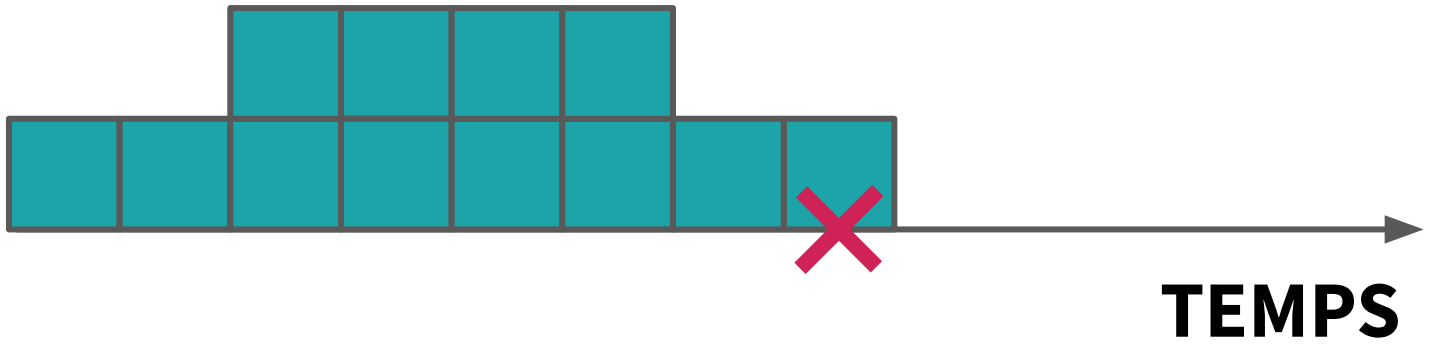
Rater une échéance !

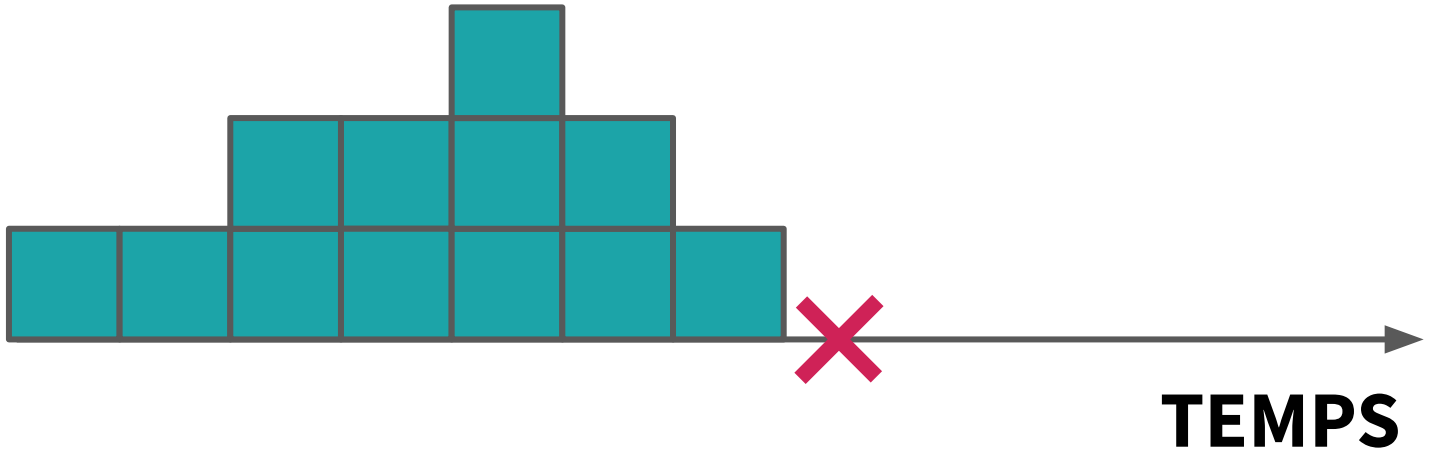


Rater une échéance ! → Conséquences dramatiques

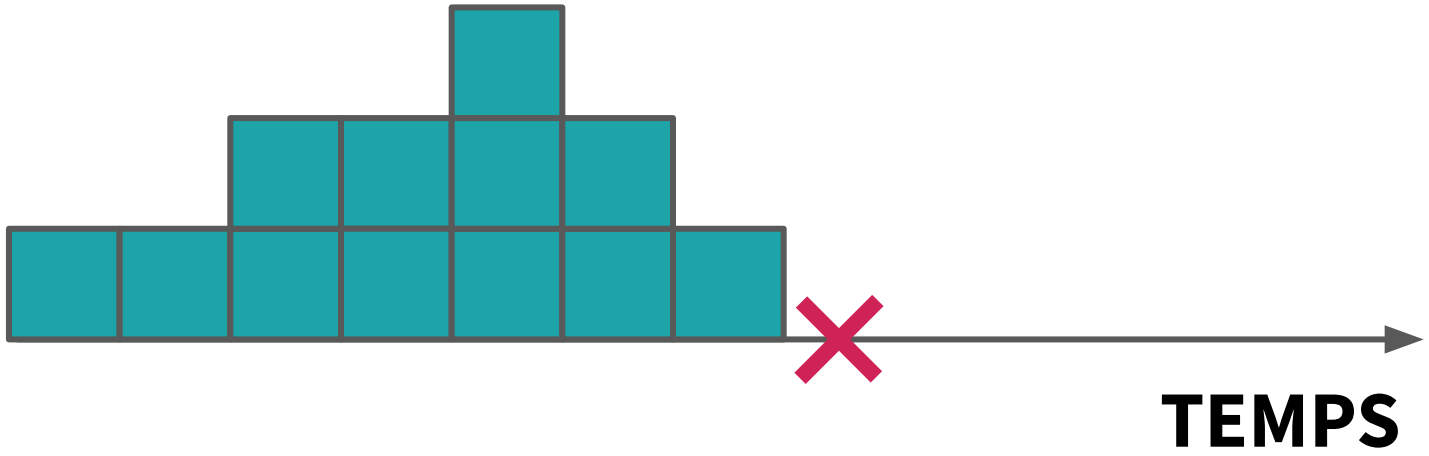


Rater une échéance ! → Conséquences dramatiques

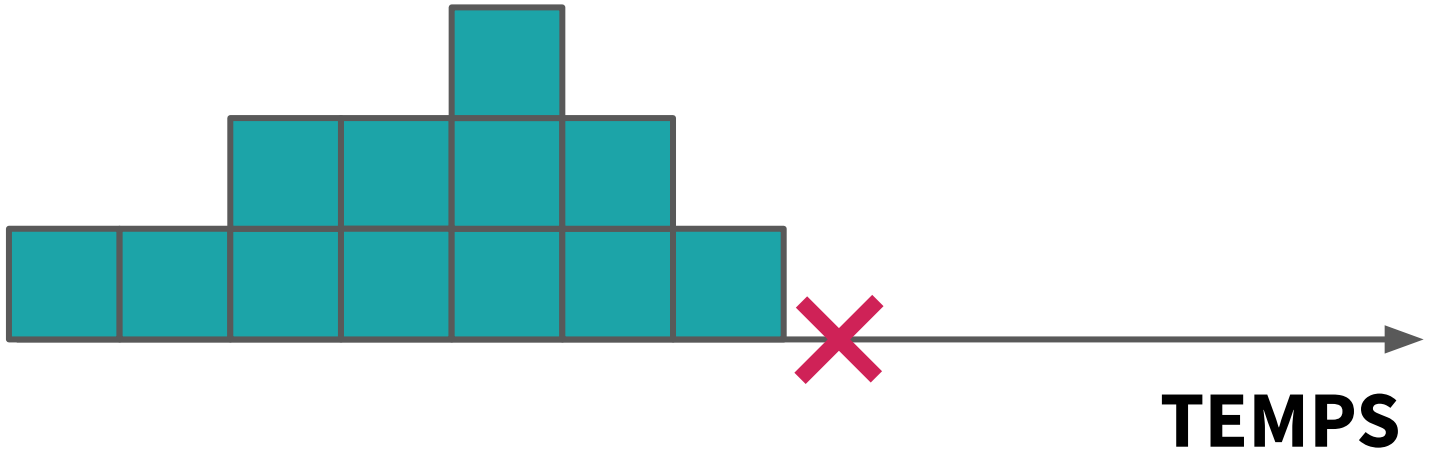


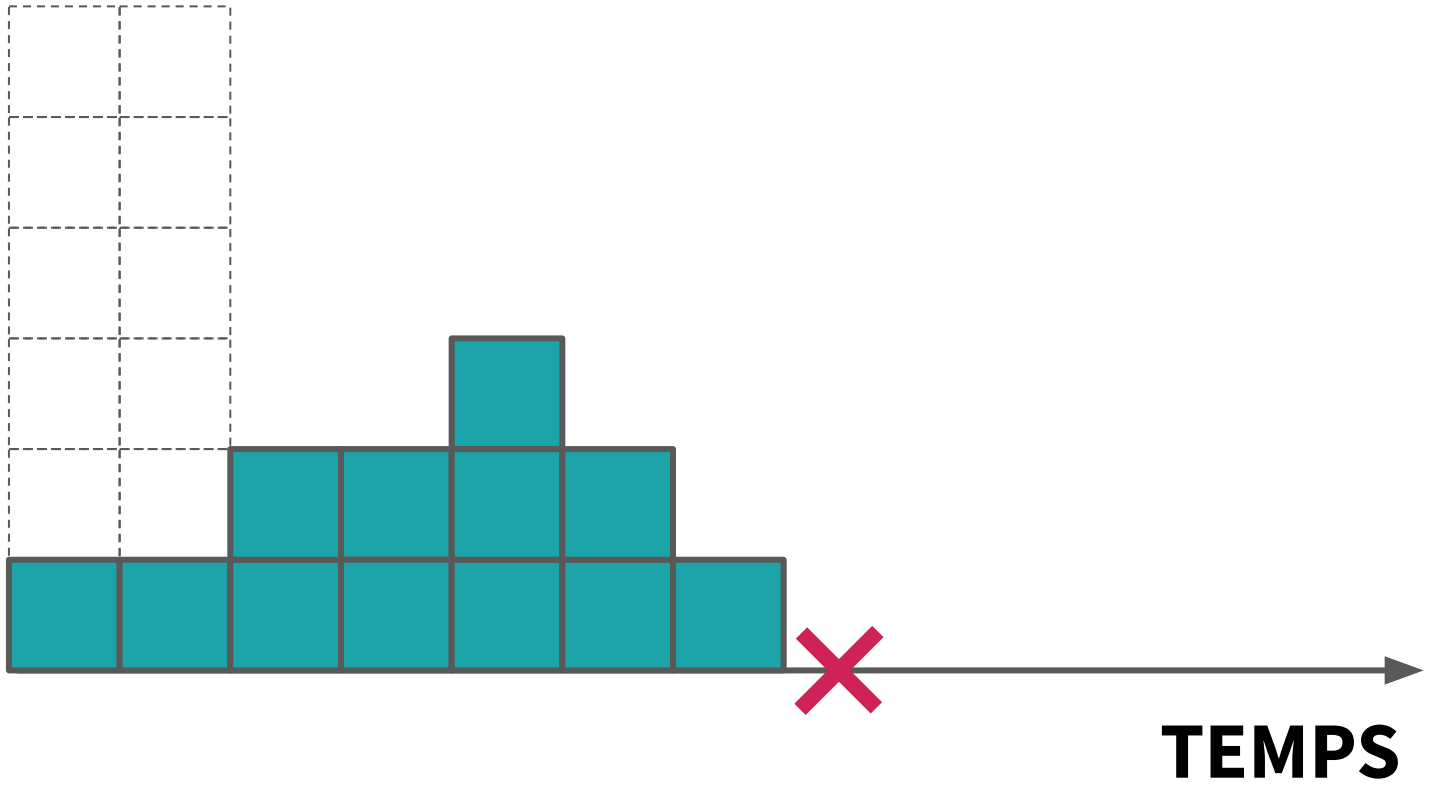

TEMPS

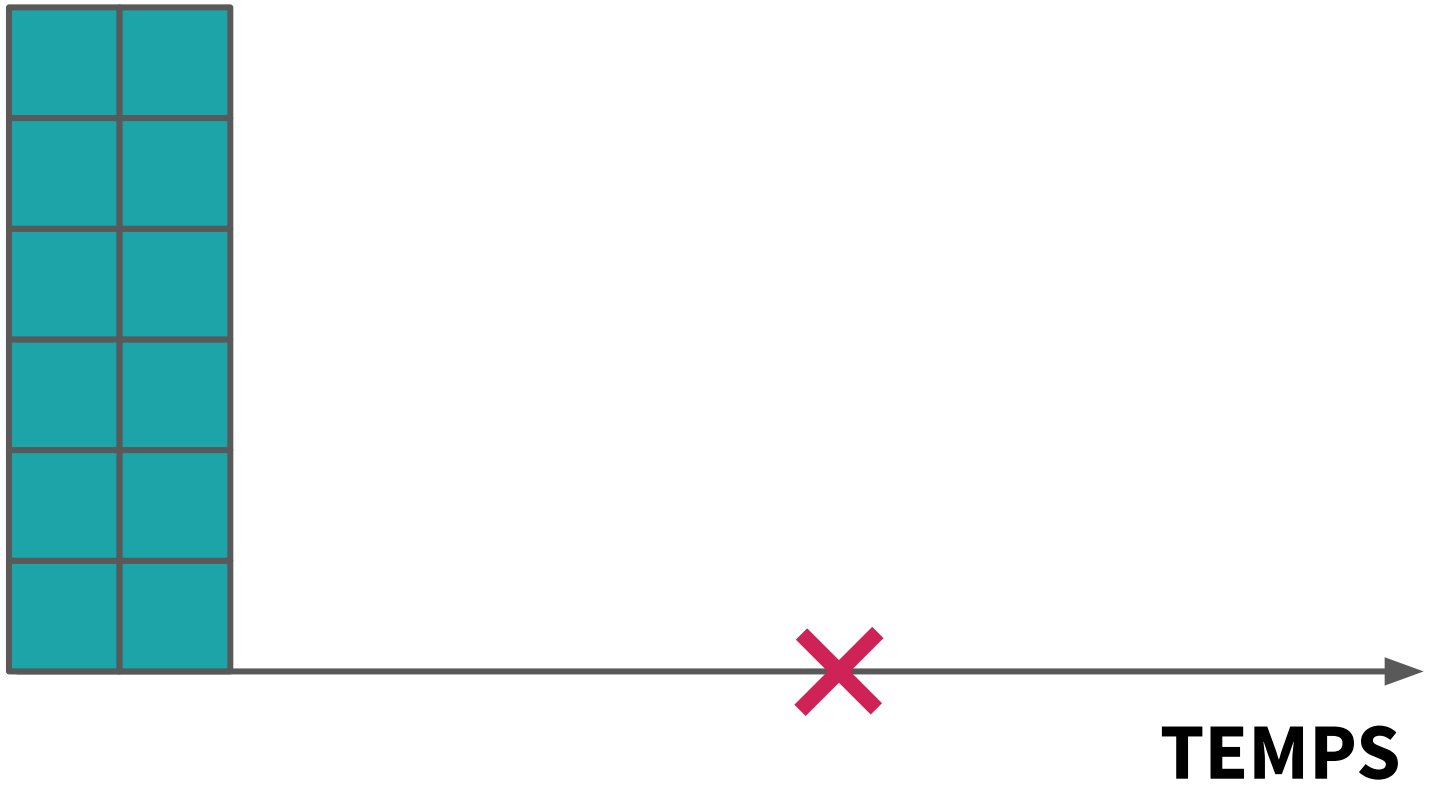
Energie dépensée avant l’échéance → tout ok.

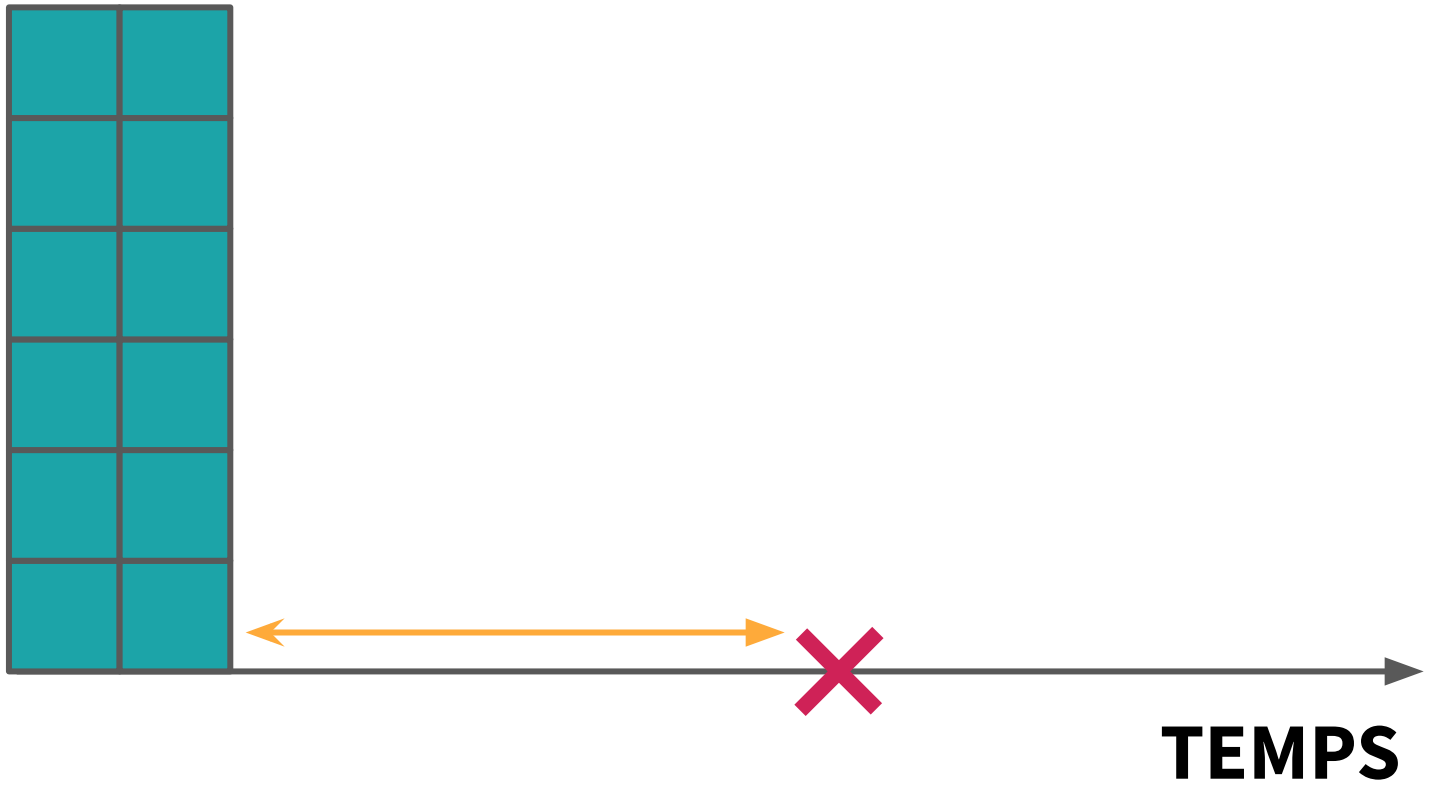


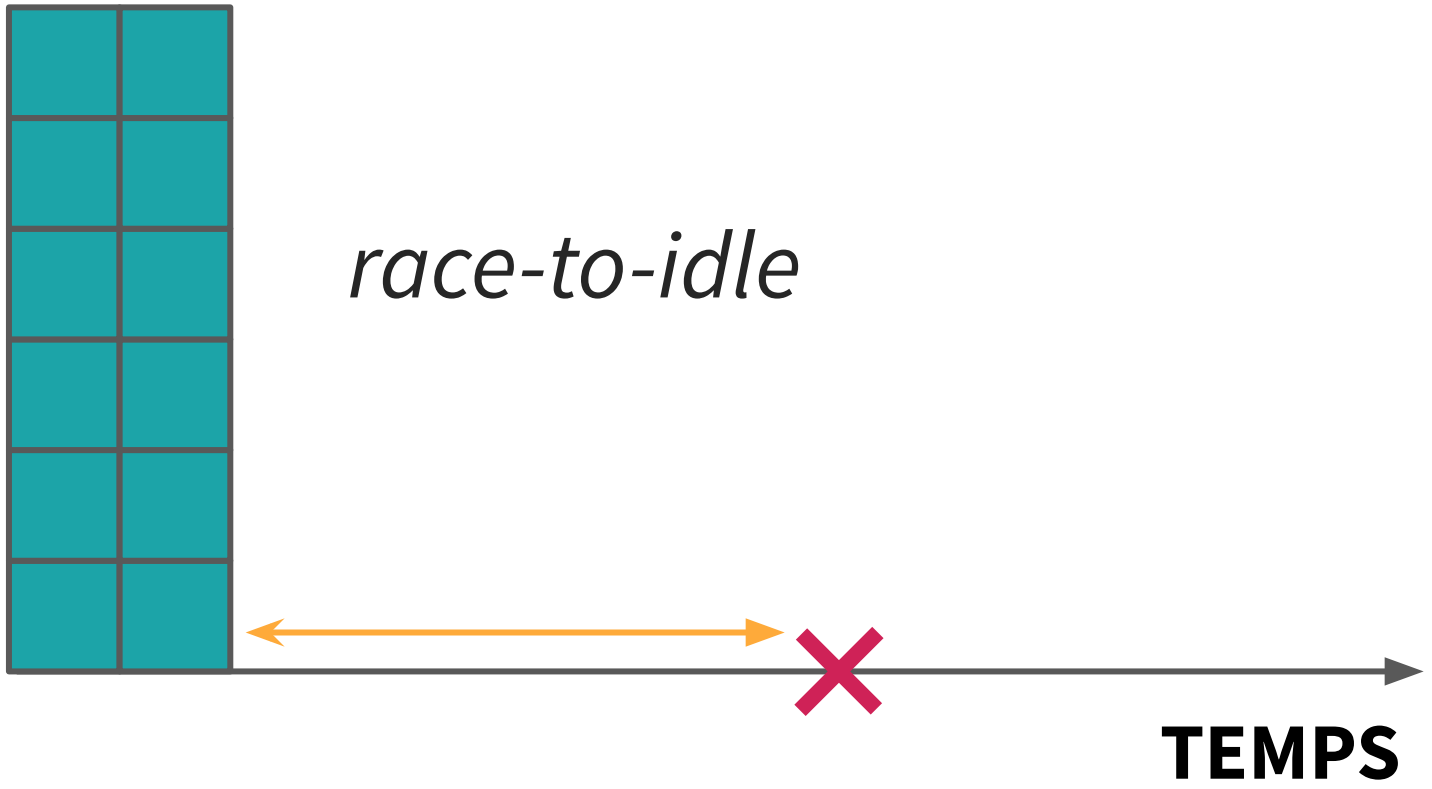
Energie dépensée avant l’échéance → tout ok ?




TEMPS

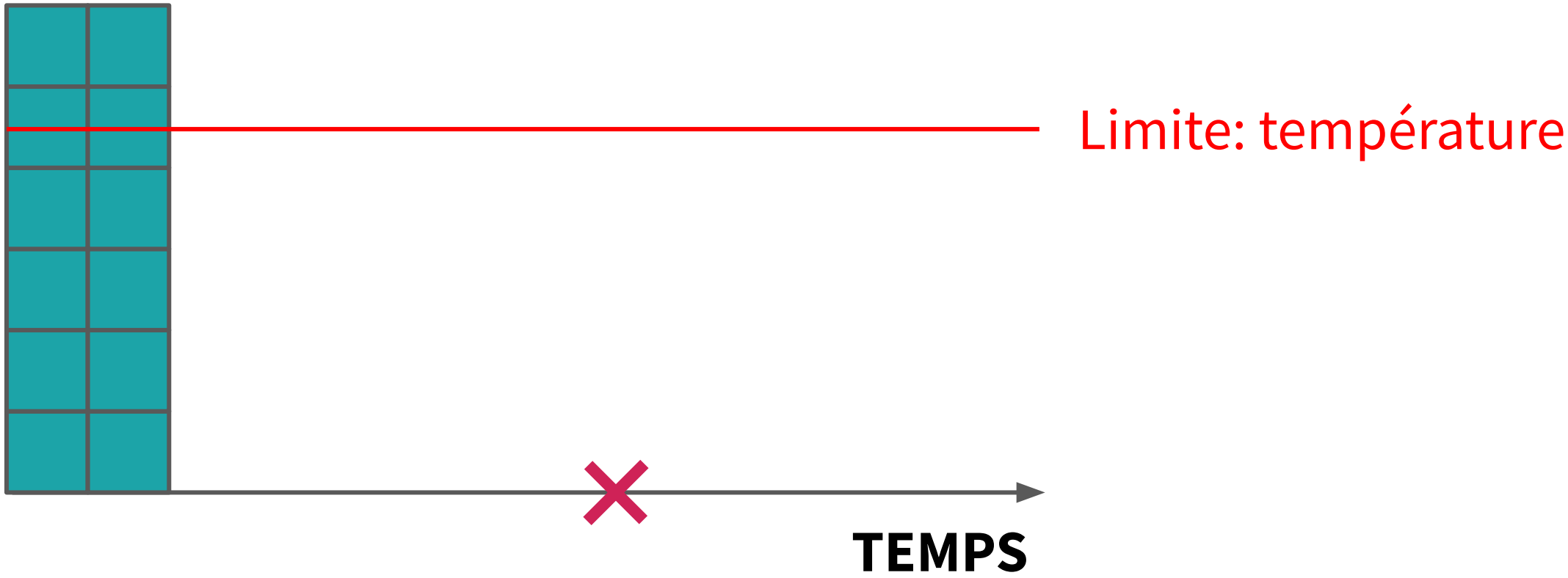

TEMPS

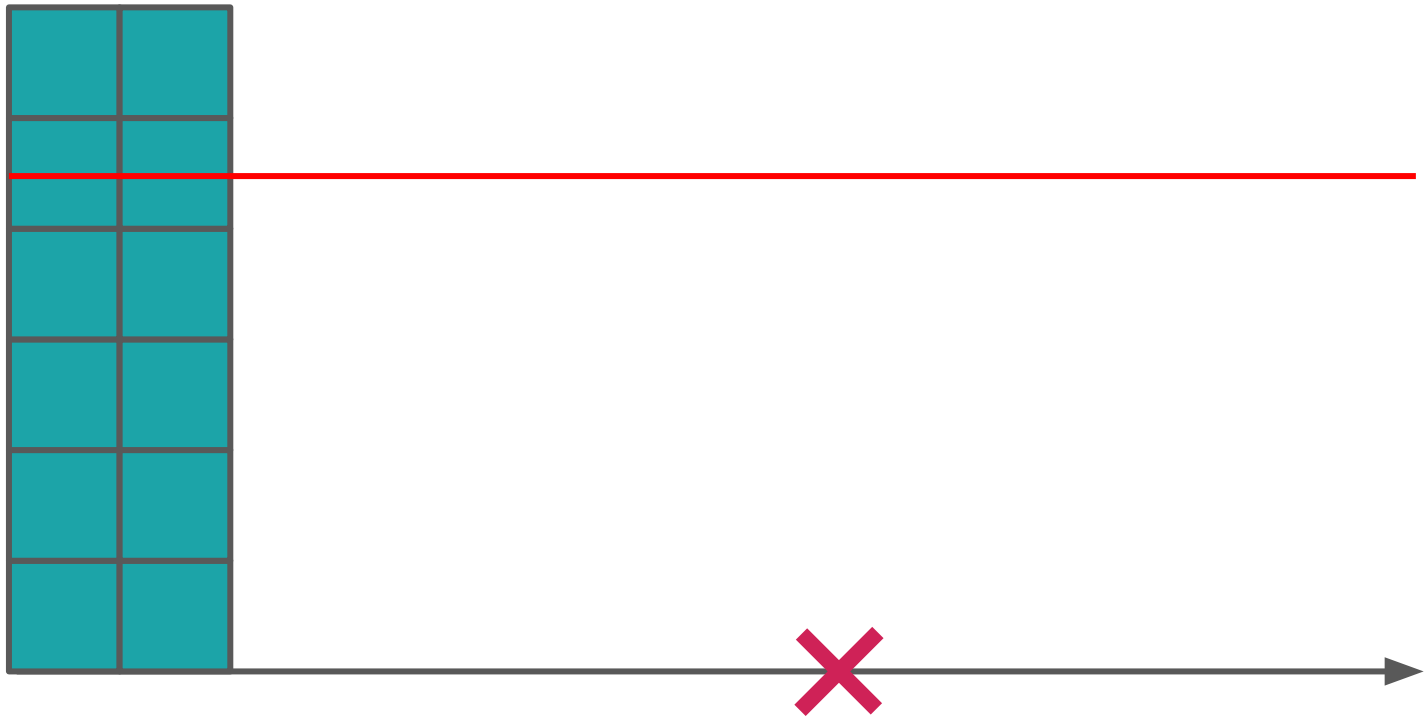

TEMPS

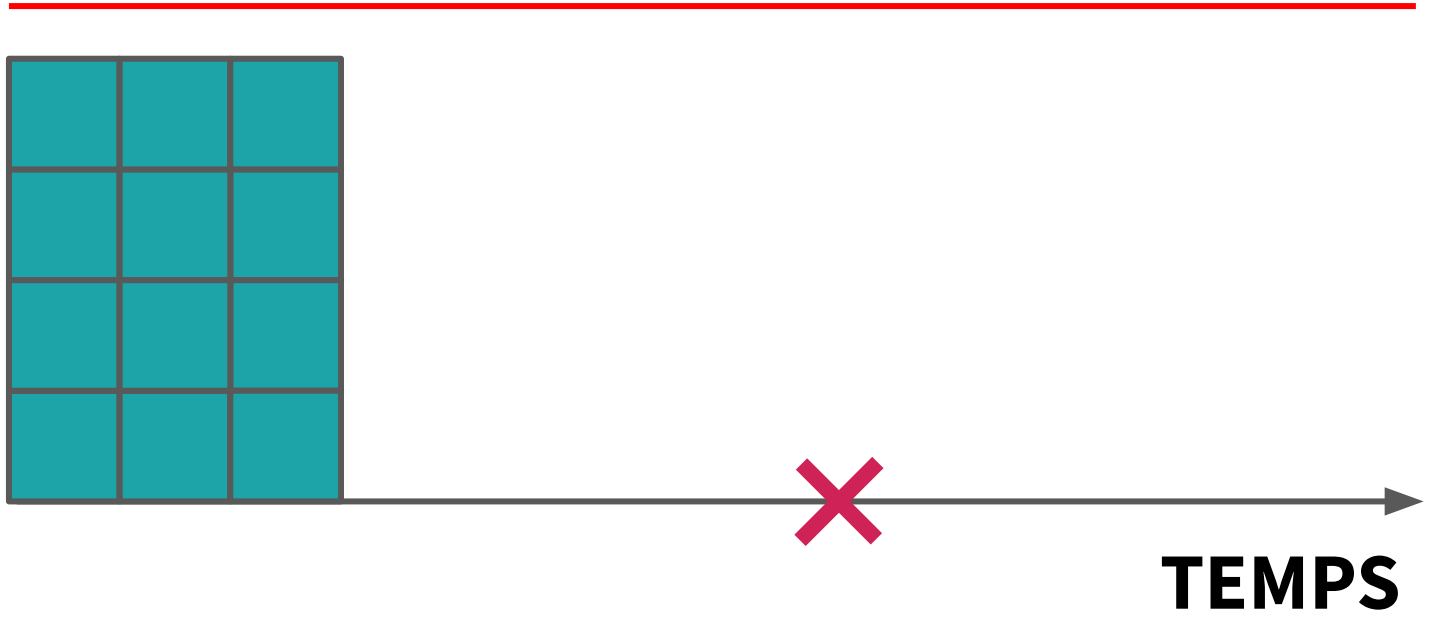

race-to-idle
TEMPS

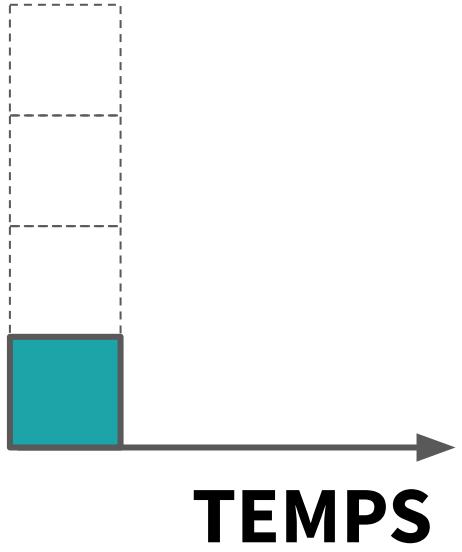

TEMPS

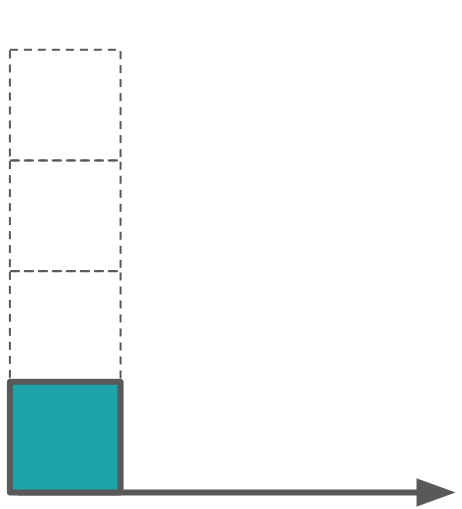

TEMPS


Limite: température
TEMPS

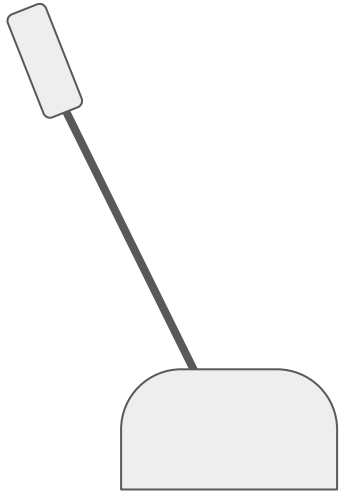

Limite: température
TEMPS

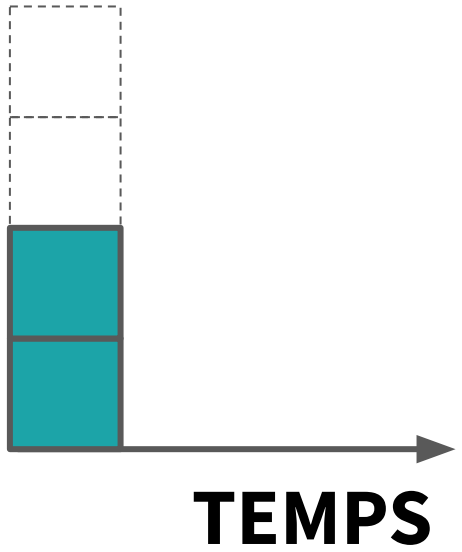

TEMPS

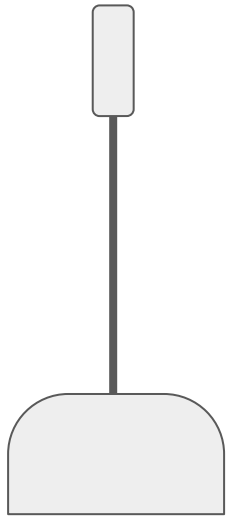

TEMPS

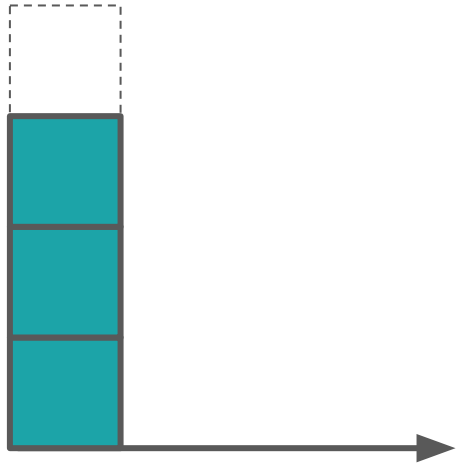


TEMPS

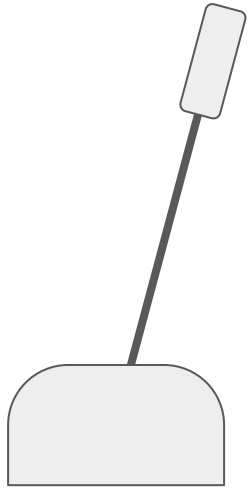


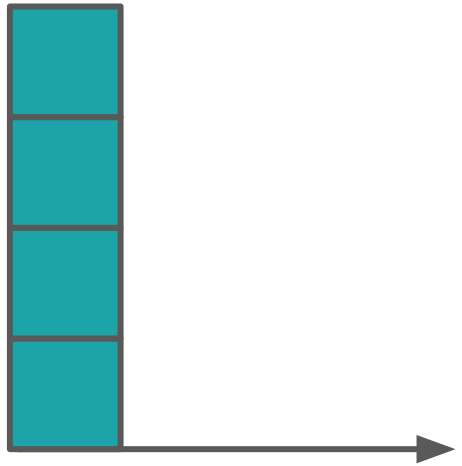

TEMPS



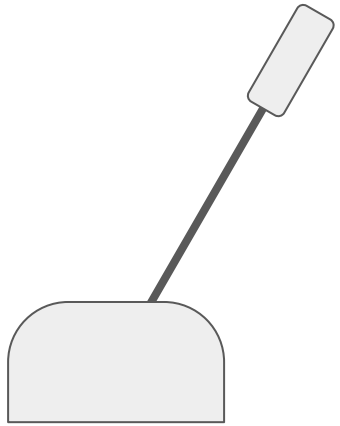


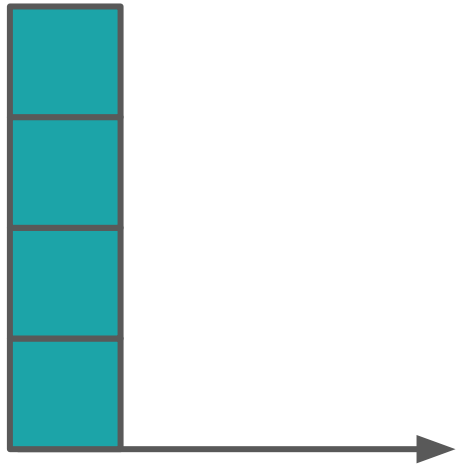
TEMPS



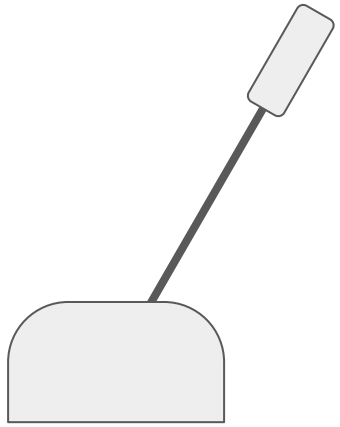


TEMPS

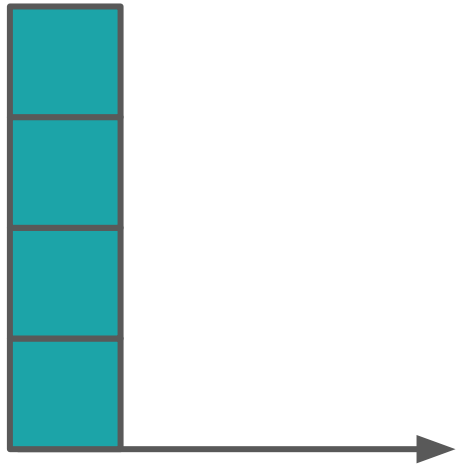




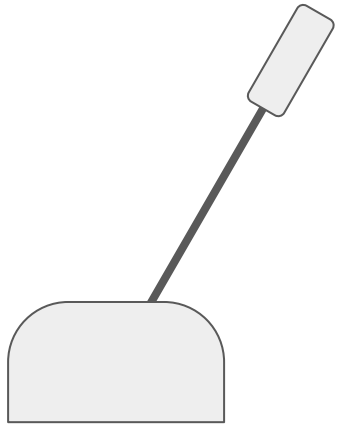
TEMPS



Vitesse

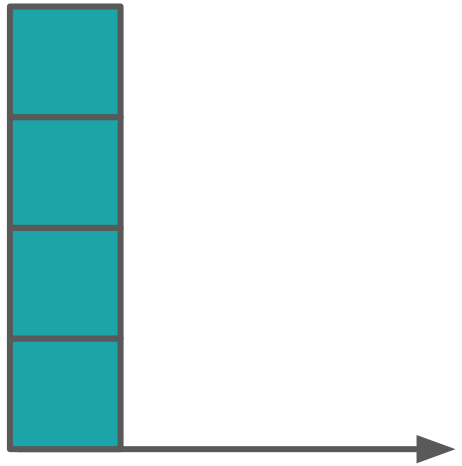


TEMPS

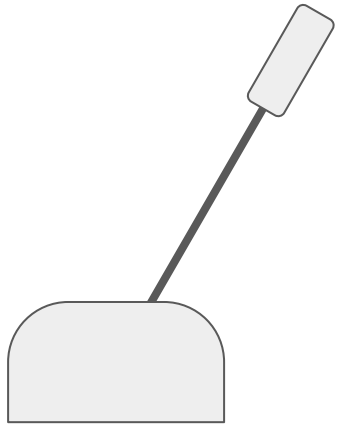


~~Vitesse~~

Fréquence

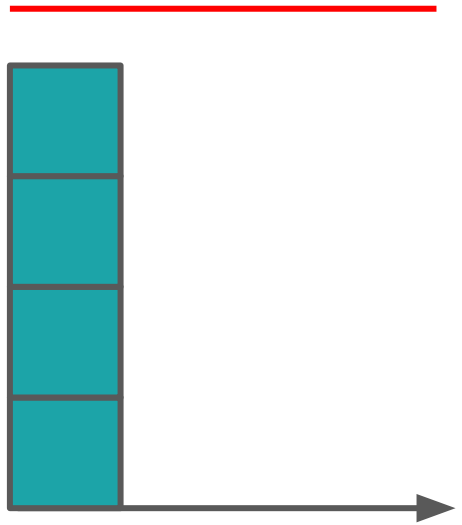


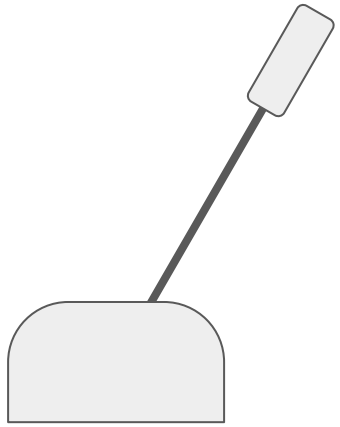
TEMPS



~~Vitesse~~

Fréquence
(MHz, GHz, etc.)





~~Vitesse~~

Fréquence
(MHz, GHz, etc.)

W

ENERGIE

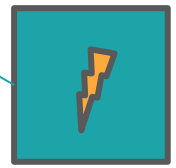
TRAVAIL

≃

W

ENERGIE **TRAVAIL**

Coût

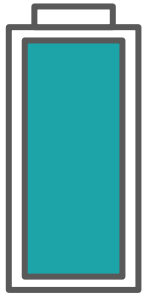


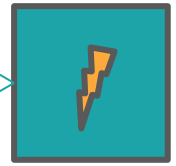
≃

W

ENERGIE **TRAVAIL**

Coût



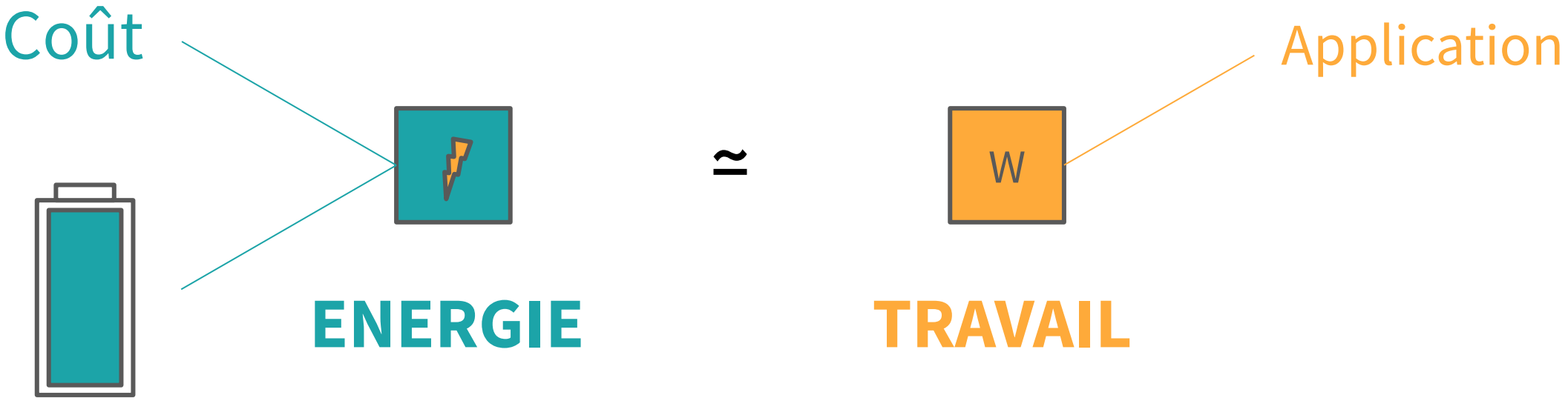


ENERGIE

≃

W

TRAVAIL

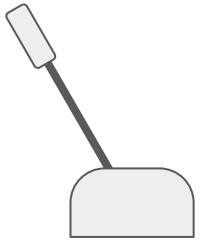

Coût
Application
≃
W
ENERGIE
TRAVAIL


Coût
≃
W
Application
Utile
ENERGIE
TRAVAIL

≃

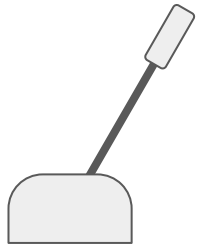
W

ENERGIE **TRAVAIL**



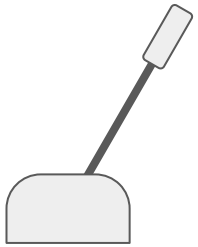
≃

ENERGIE **TRAVAIL**



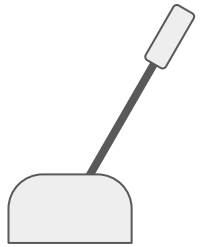
≃

ENERGIE **TRAVAIL**



≃

ENERGIE **TRAVAIL**



≃

ENERGIE **TRAVAIL**

Exemple

Objectif

Objectif

- Exécuter 10 unités de travail

Objectif

- Exécuter 10 unités de travail

Objectif

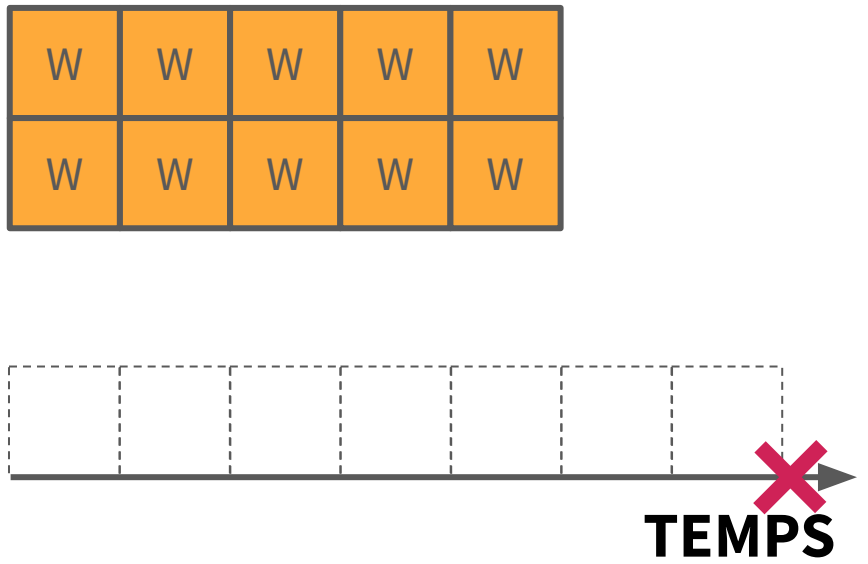
- Exécuter 10 unités de travail

- En 7 secondes

Objectif

- Exécuter 10 unités de travail

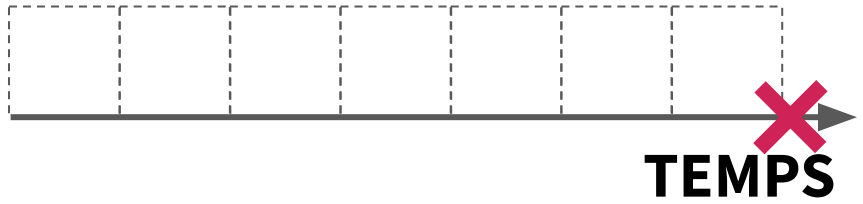
- En 7 secondes



Objectif

- Exécuter 10 unités de travail

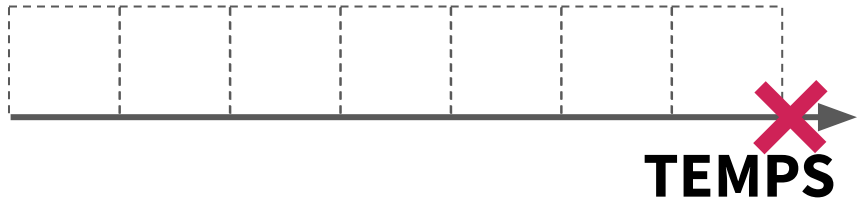
- En 7 secondes

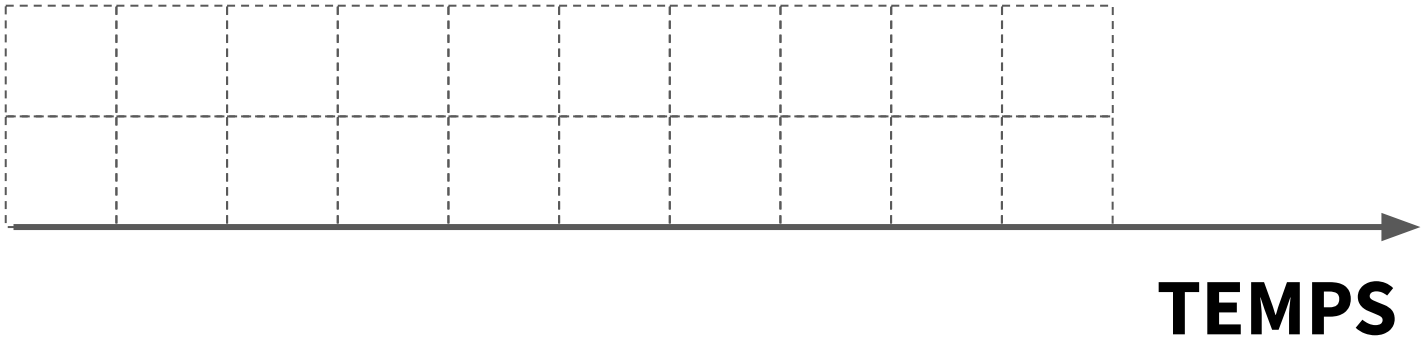


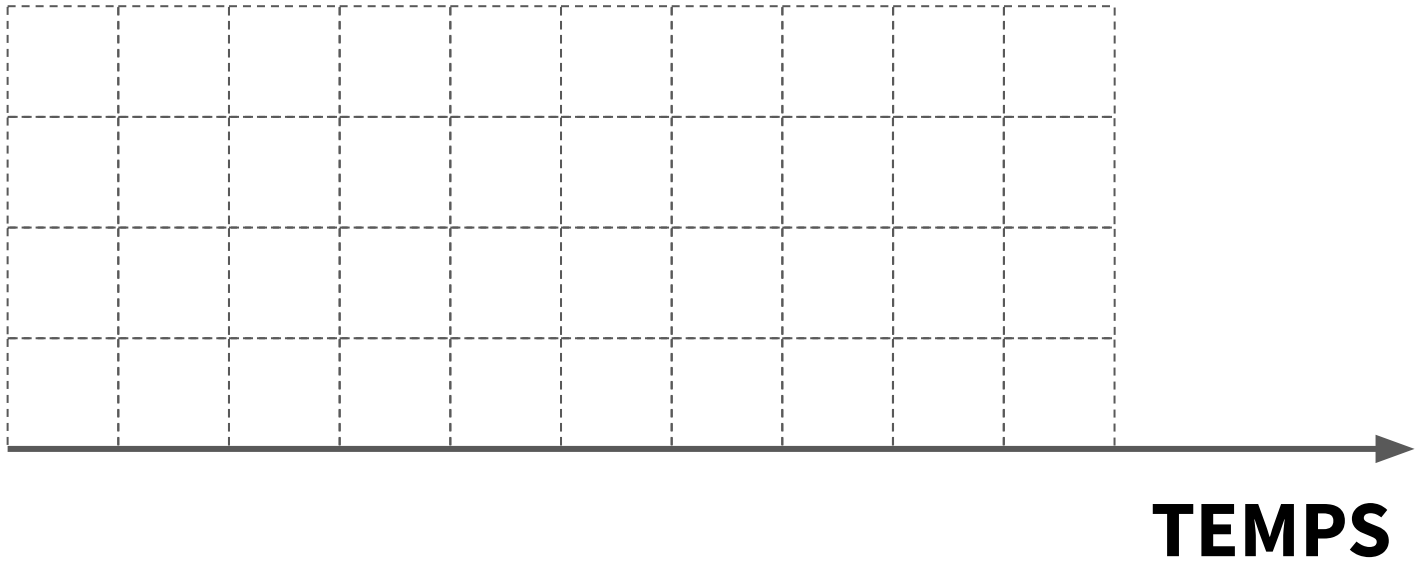
- En minimisant l’énergie consommée

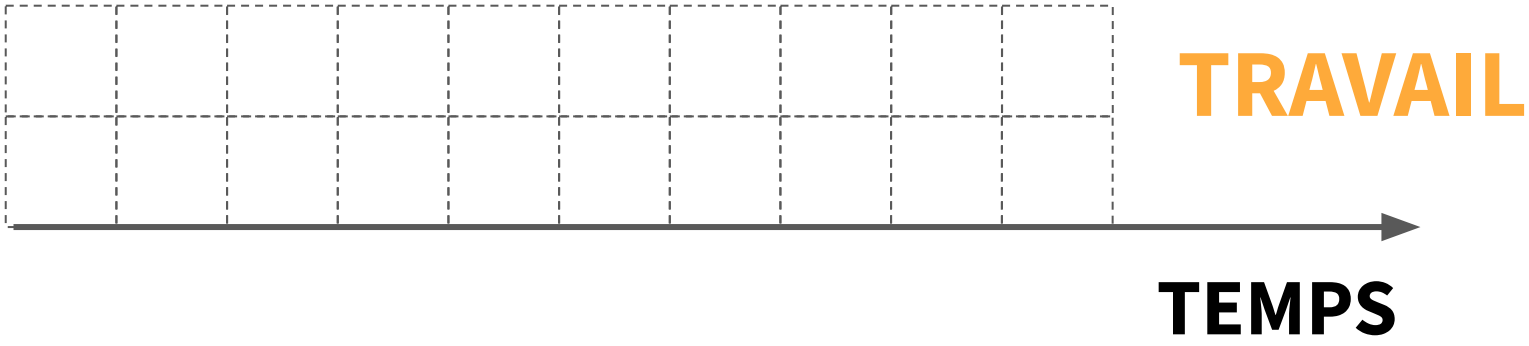
Objectif

- Exécuter 10 unités de travail
- En 7 secondes
- En minimisant l’énergie consommée

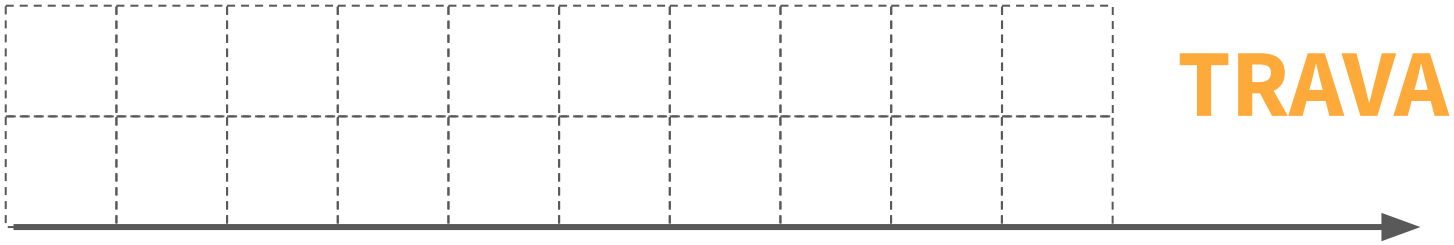



TEMPS


TEMPS

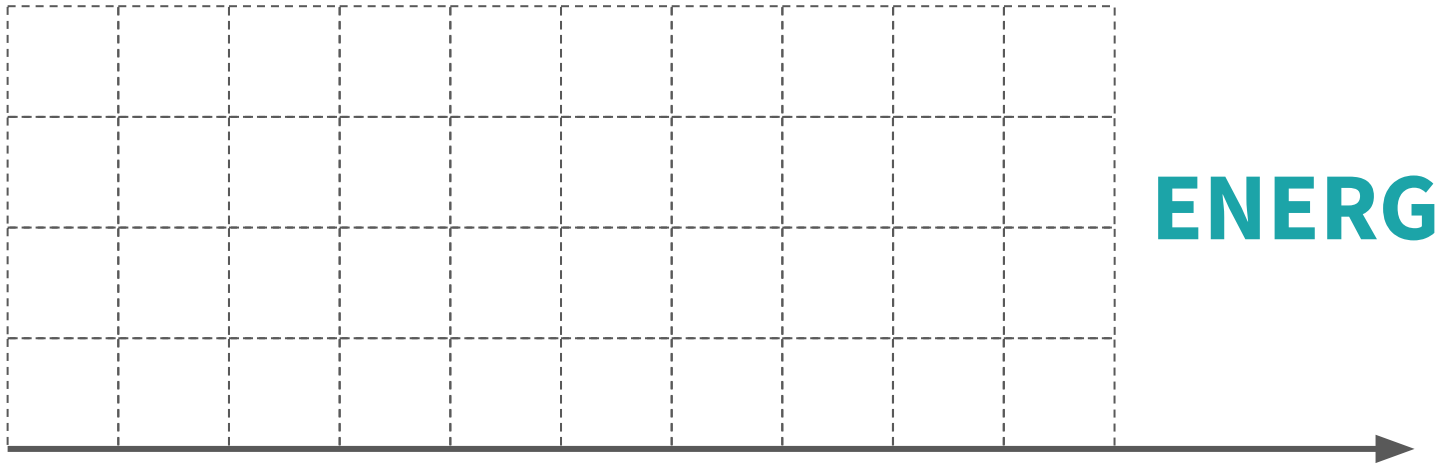

TRAVAIL
TEMPS


TEMPS



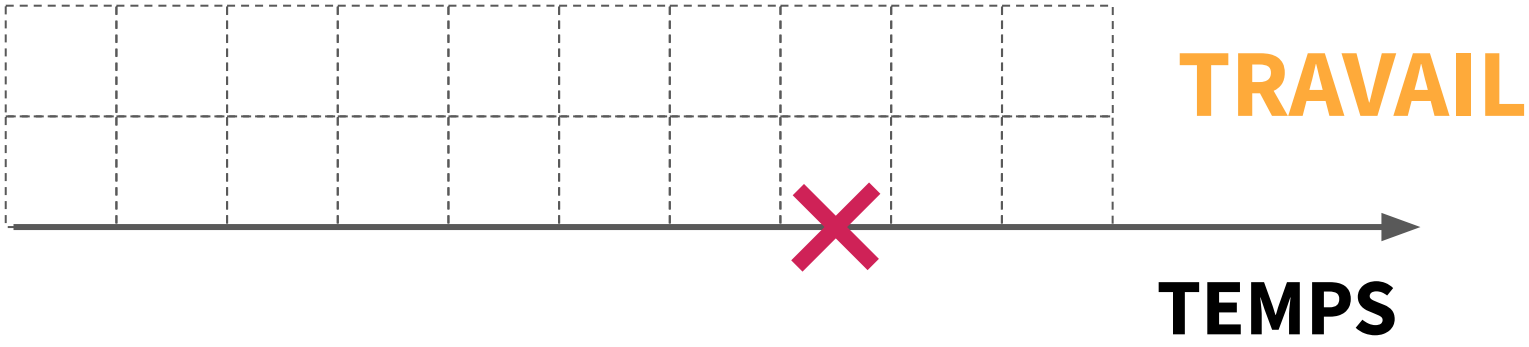
TRAVAIL

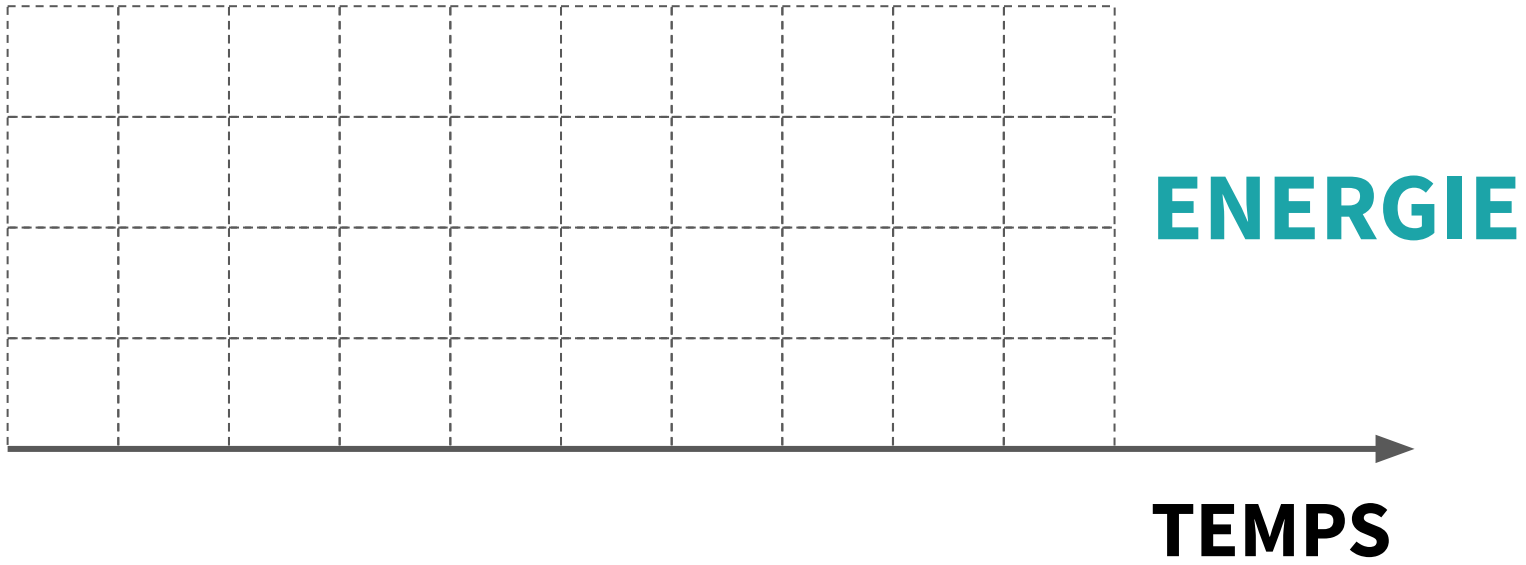
TEMPS



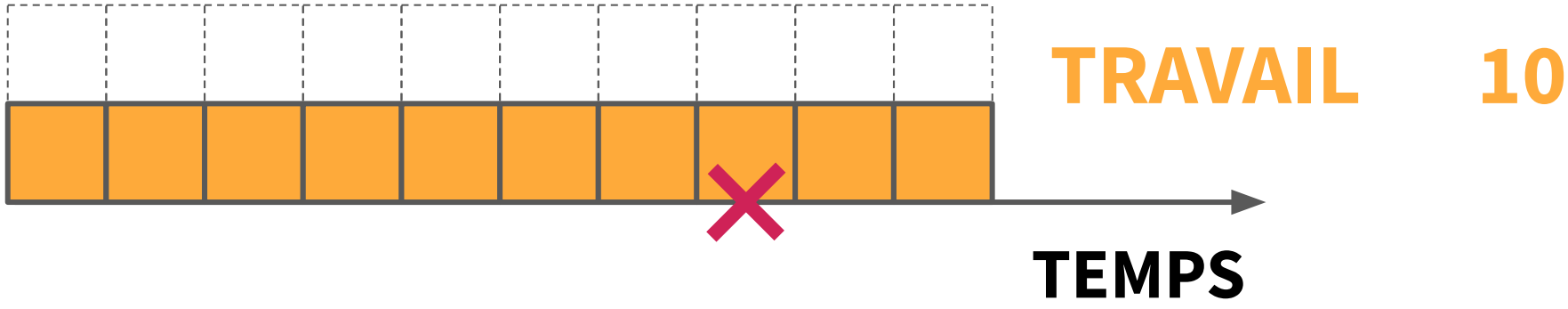
ENERGIE

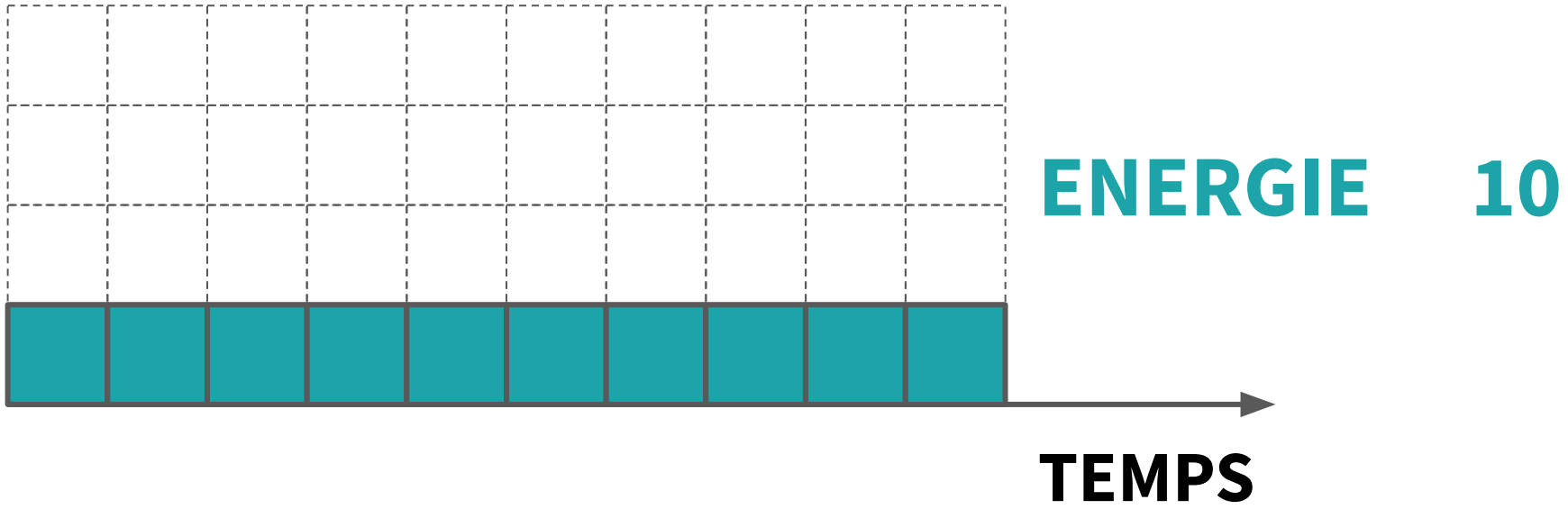
TEMPS


TRAVAIL
TEMPS

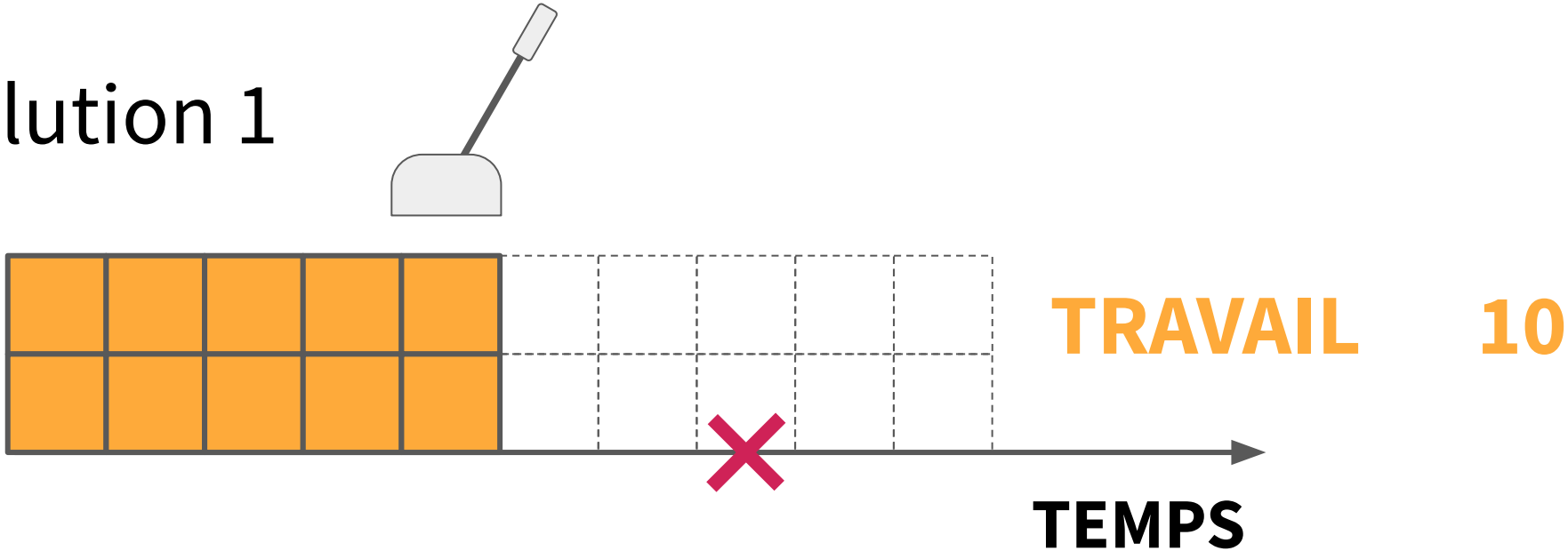

ENERGIE
TEMPS

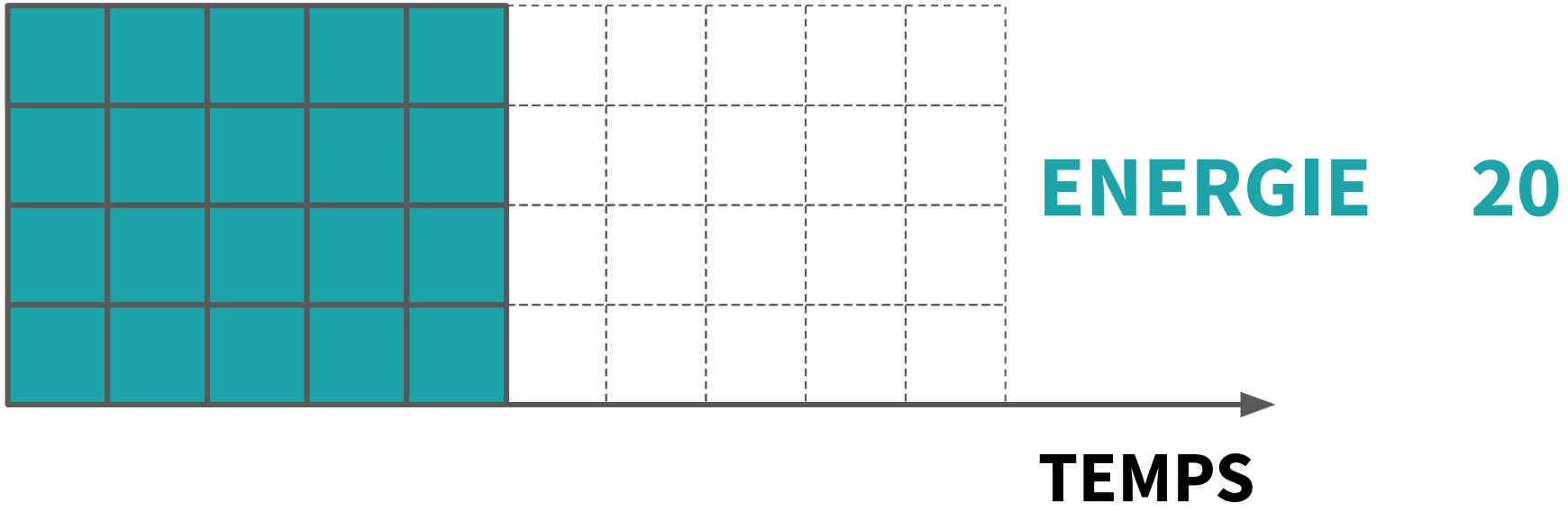
Solution 0



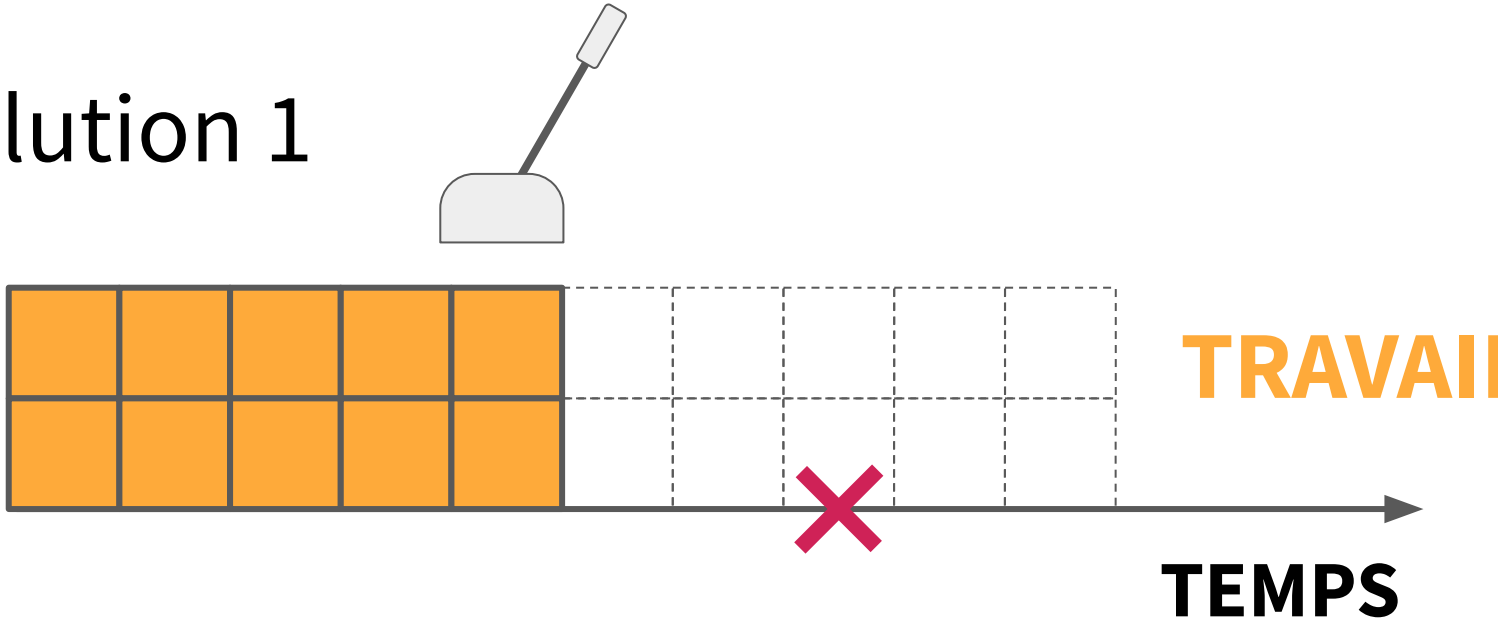


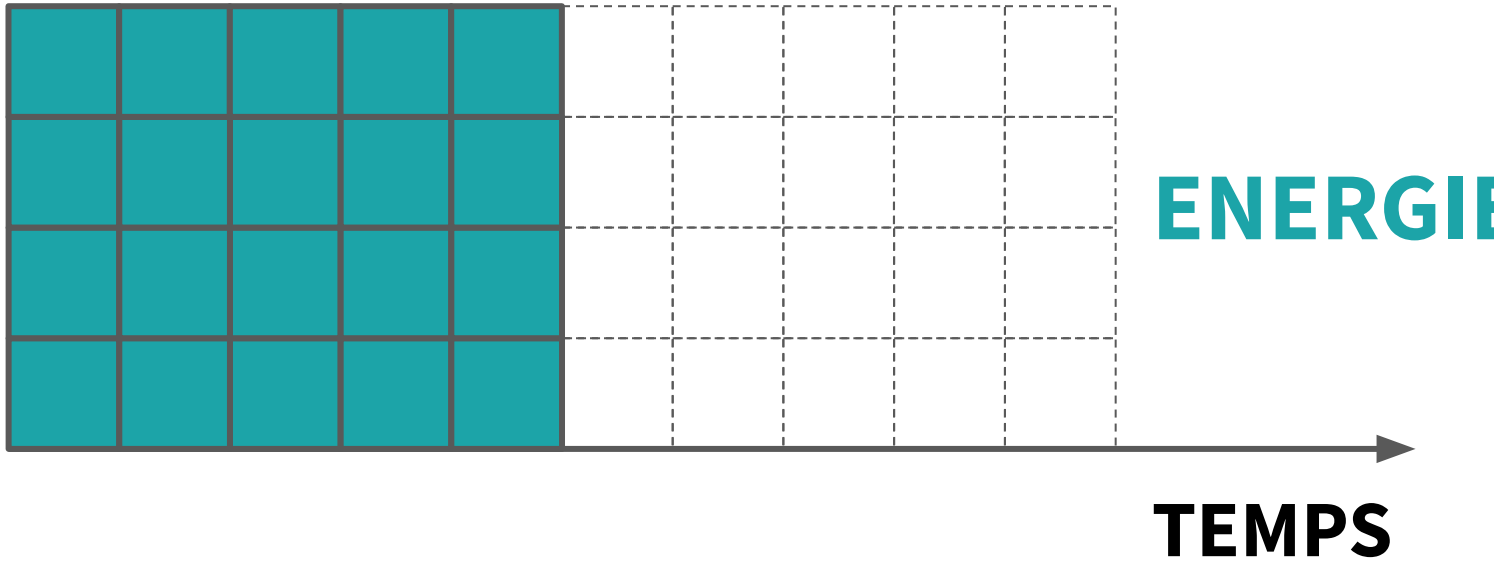
Solution 1



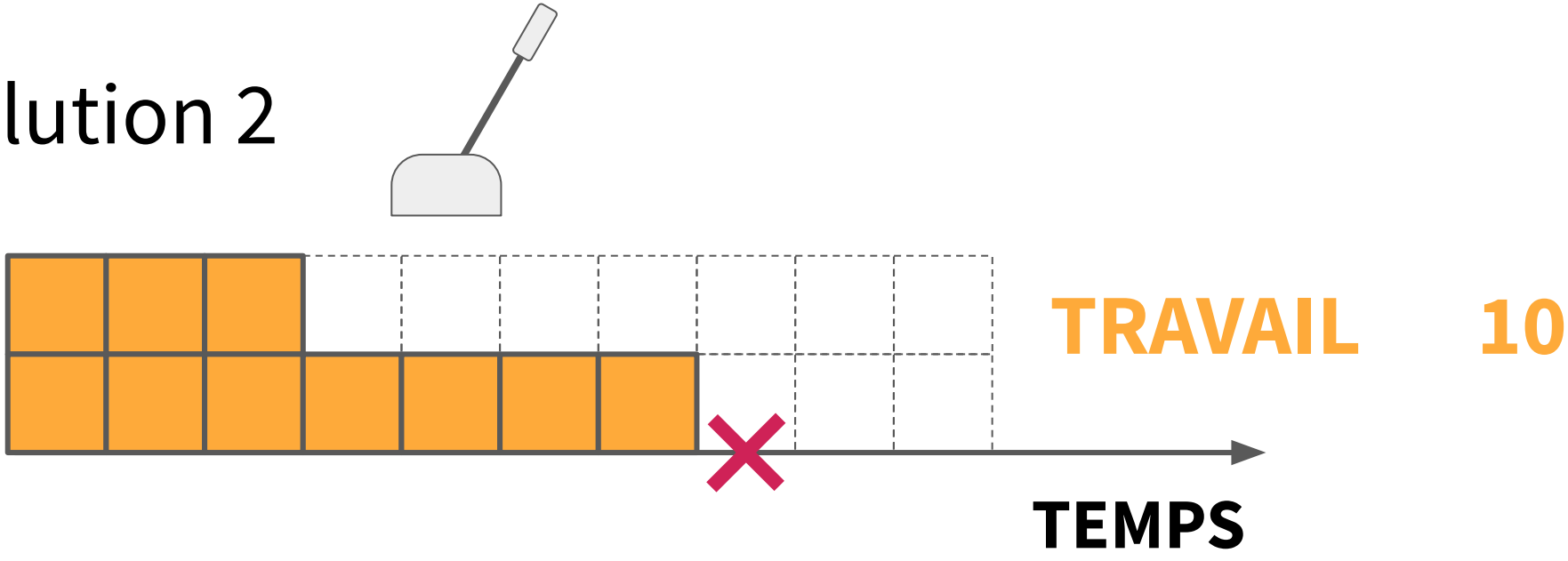


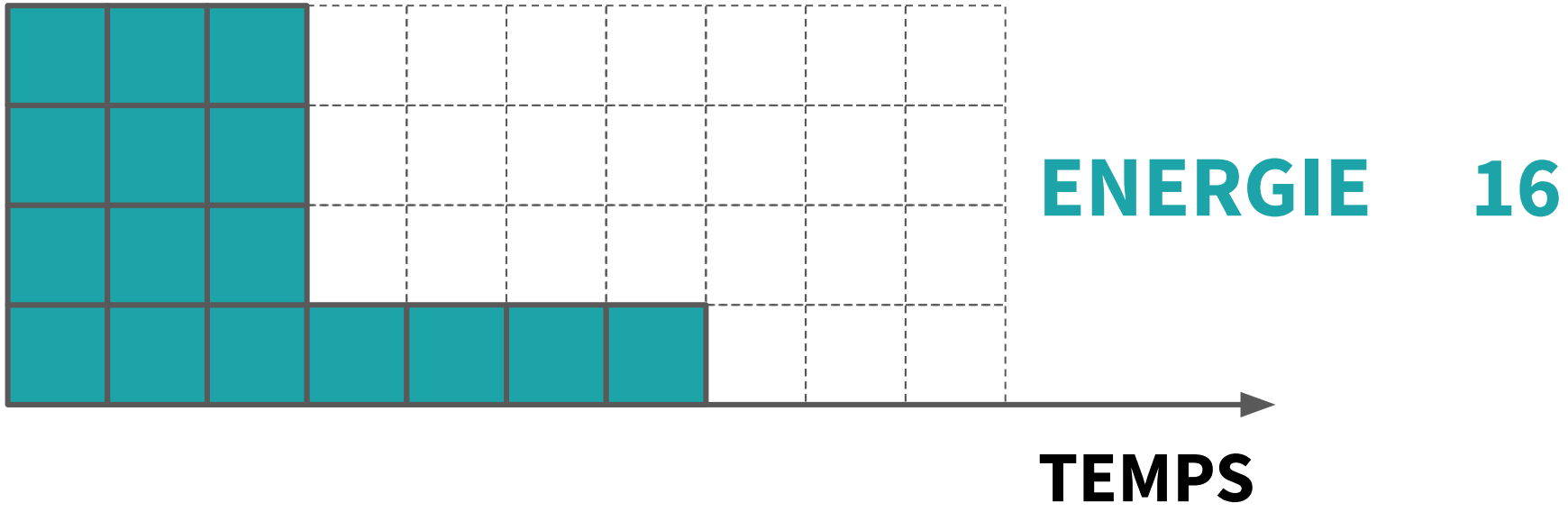
Solution 1



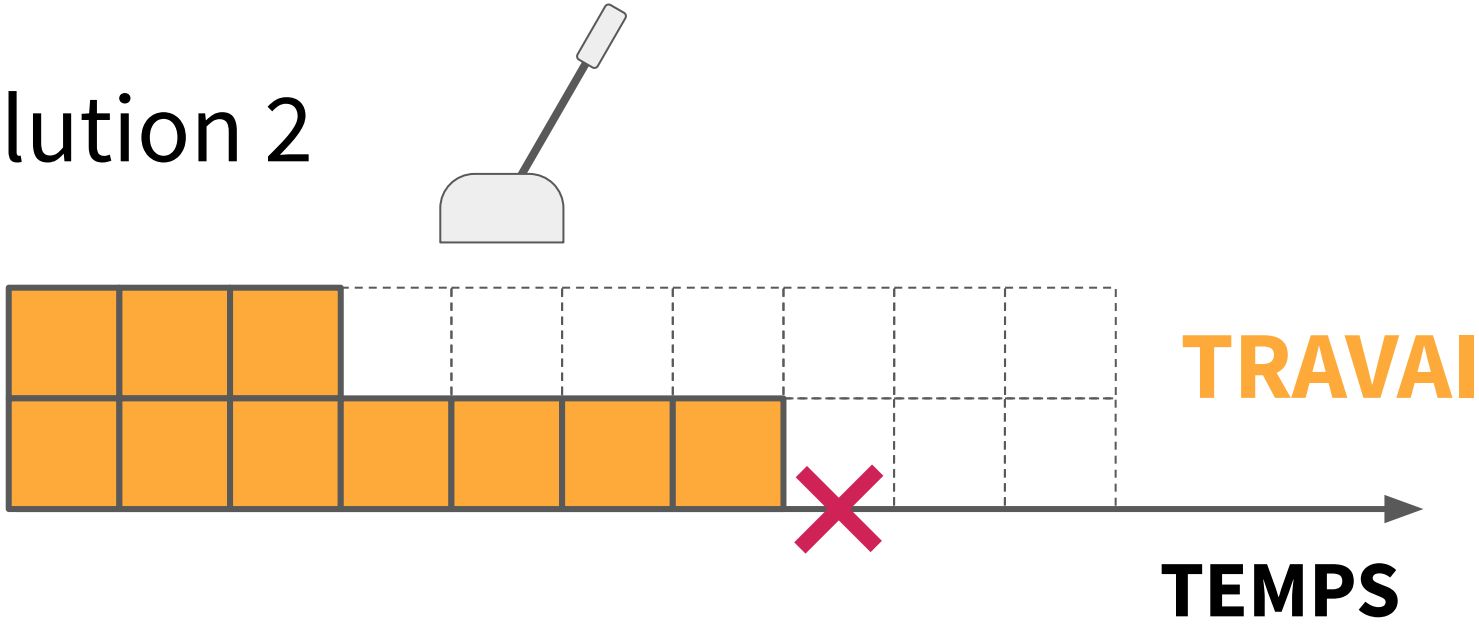


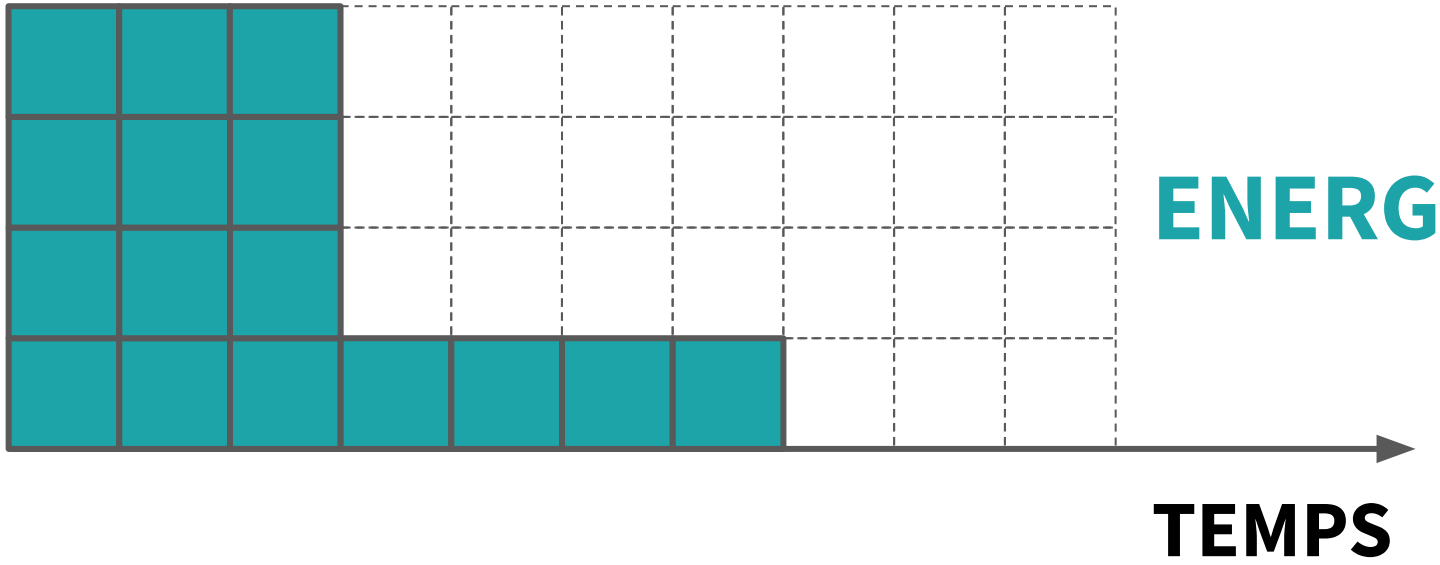
Solution 2





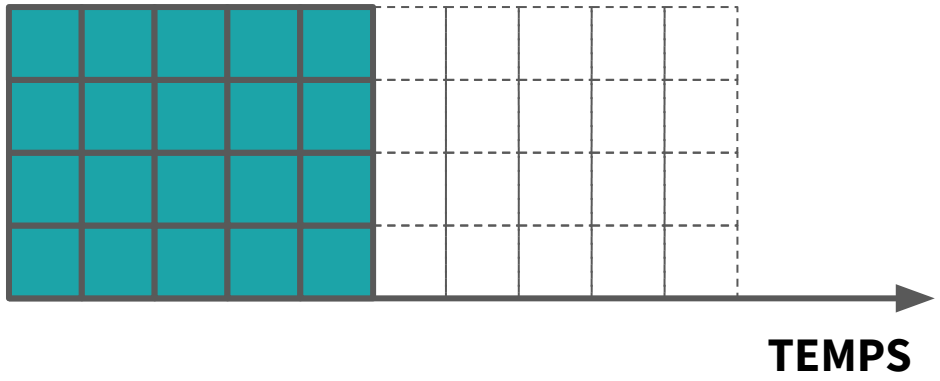
Solution 2





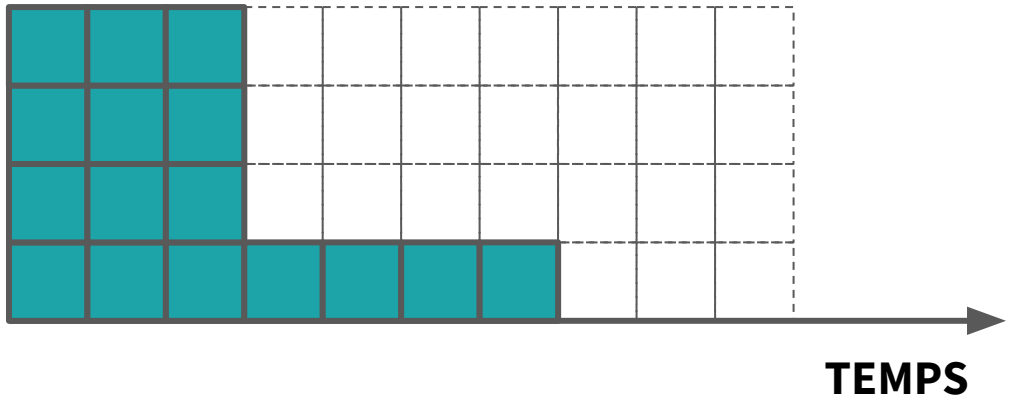
Solutions 1 et 2, facteur quadratique

1



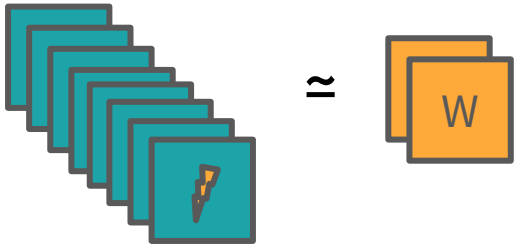
ENERGIE 20 (x2 !)

2

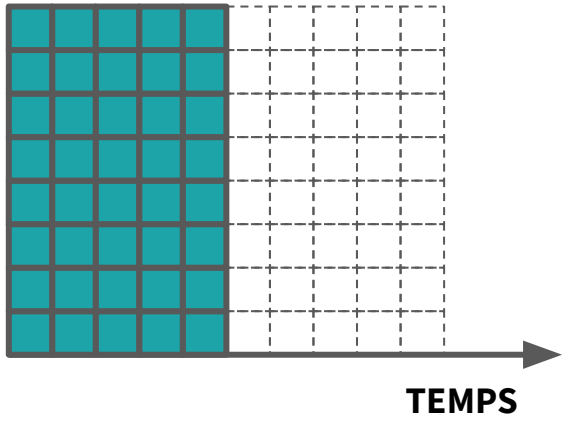


ENERGIE 16 (-20%)

Solutions 1 et 2, facteur cubique

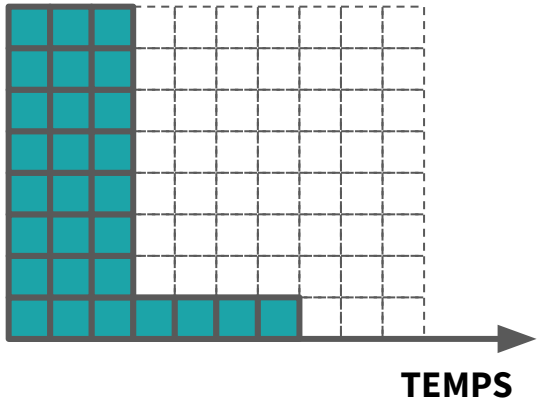


1

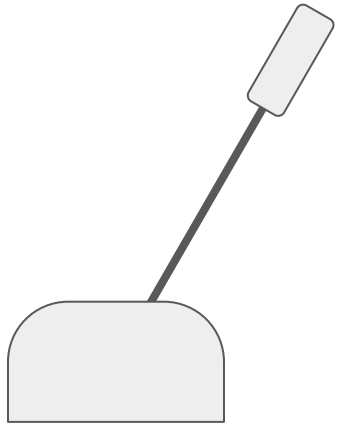


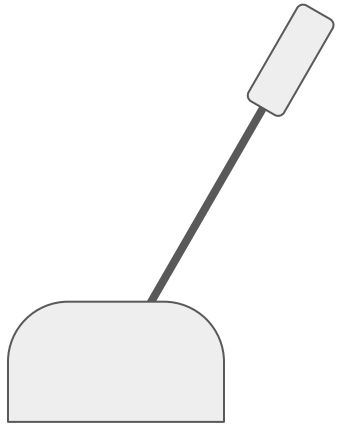
ENERGIE 40 (x4 !)

2

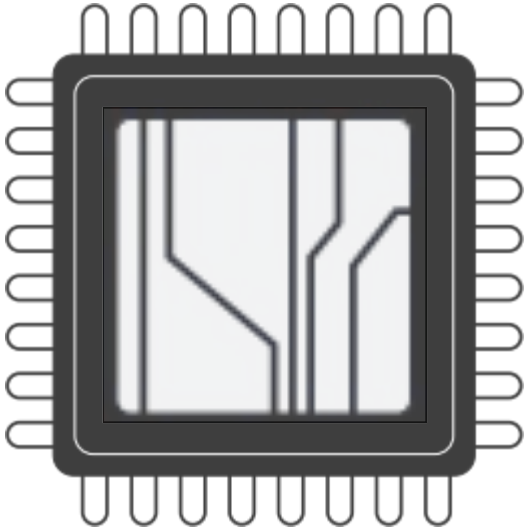


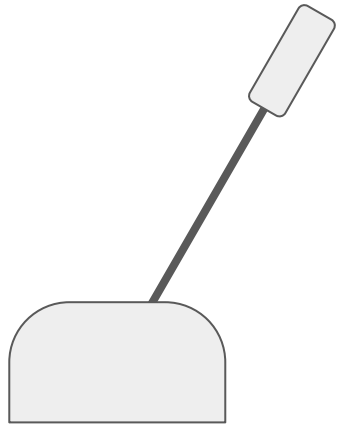
ENERGIE 28 (-30%)

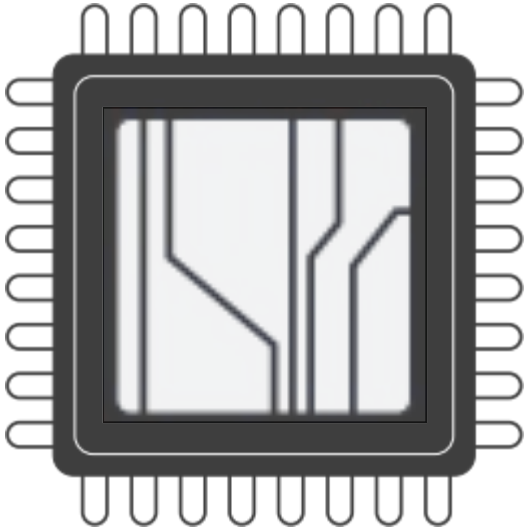




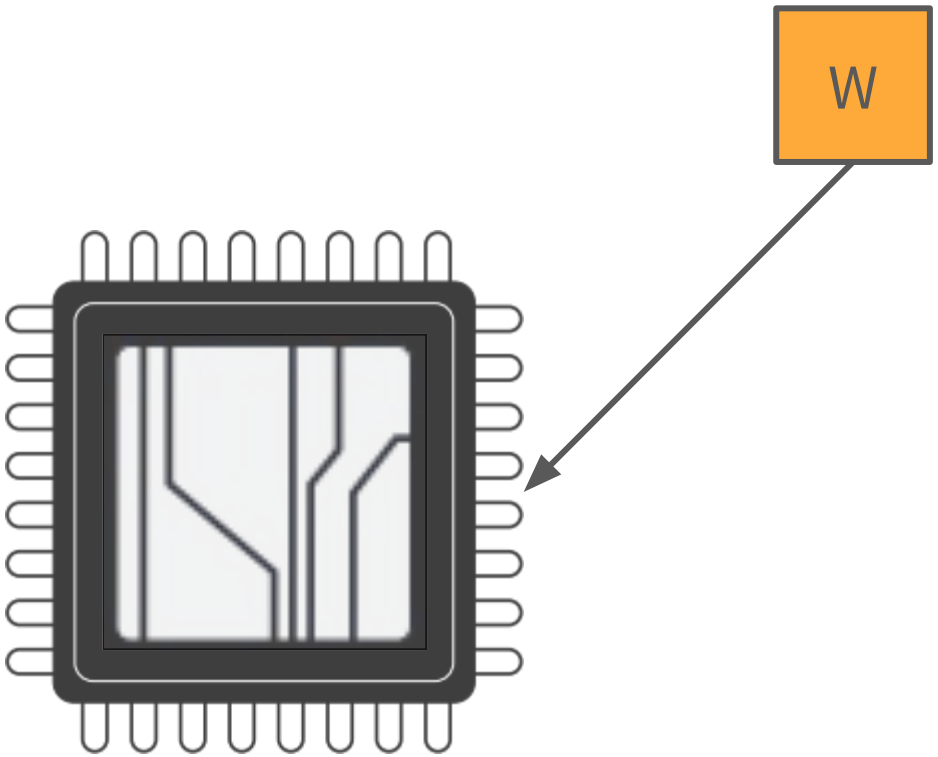

ANDROID

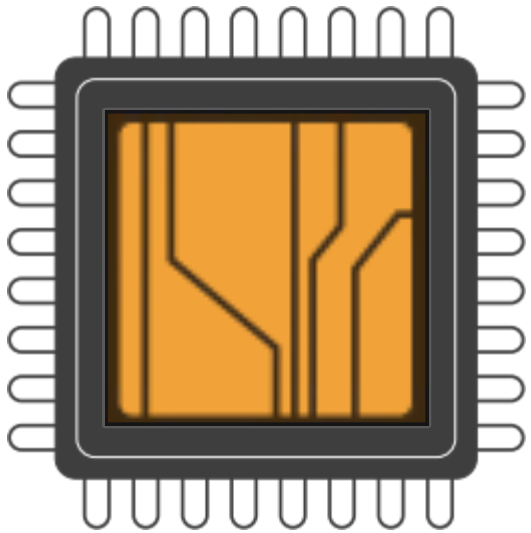


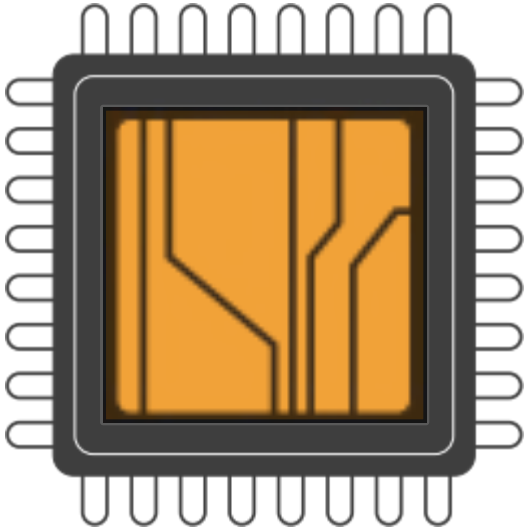


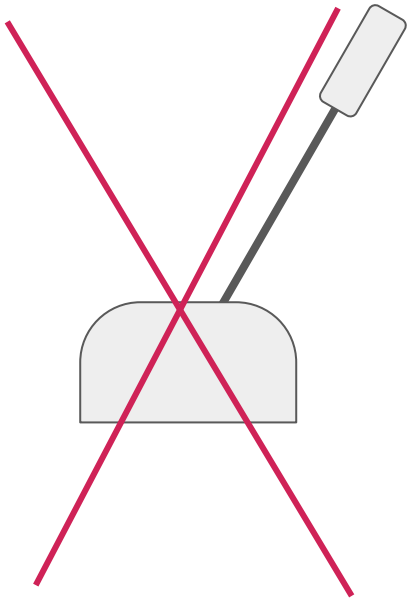


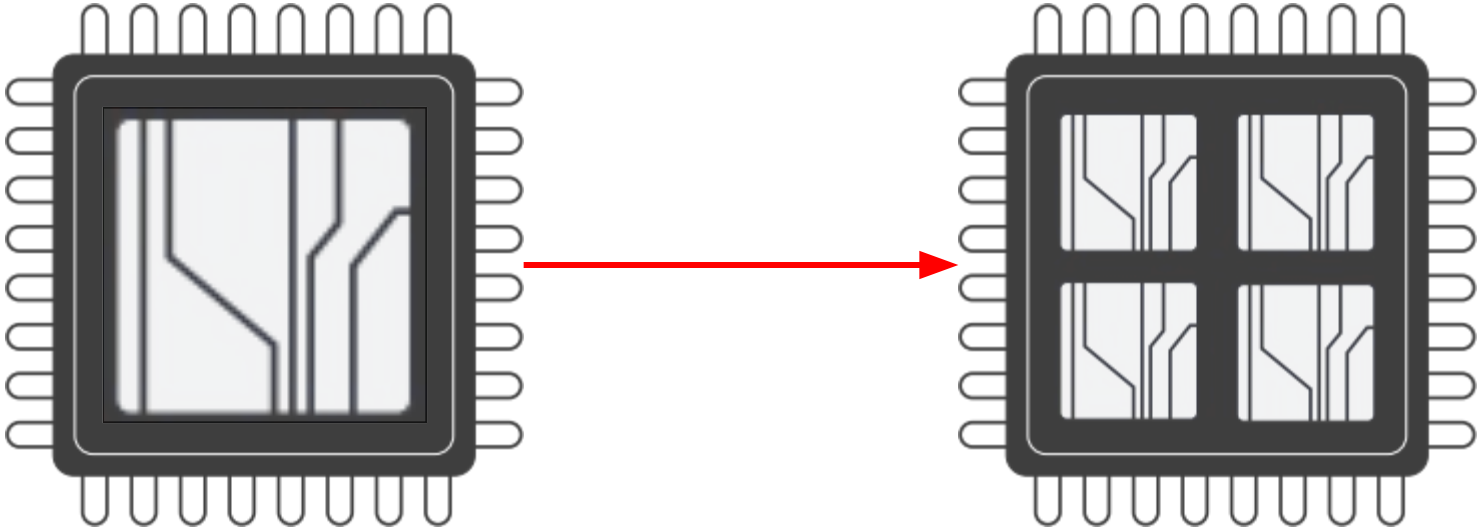
W ?

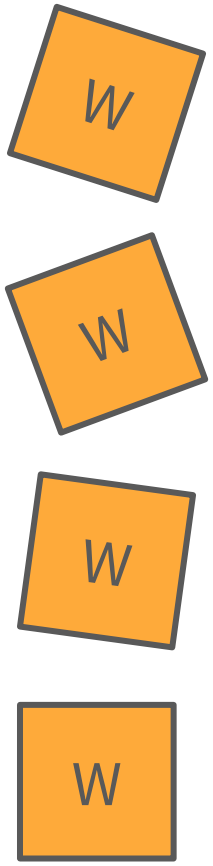

W

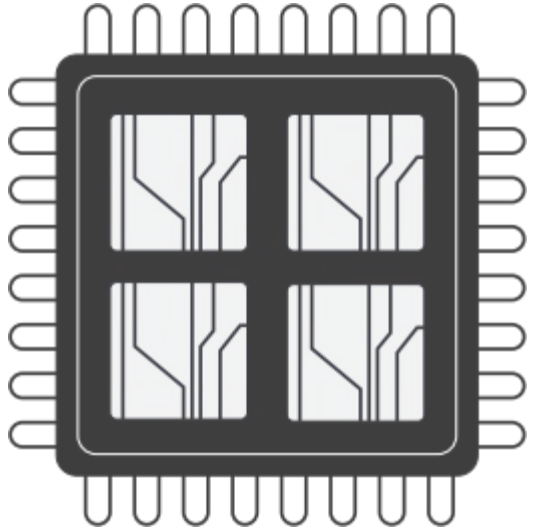


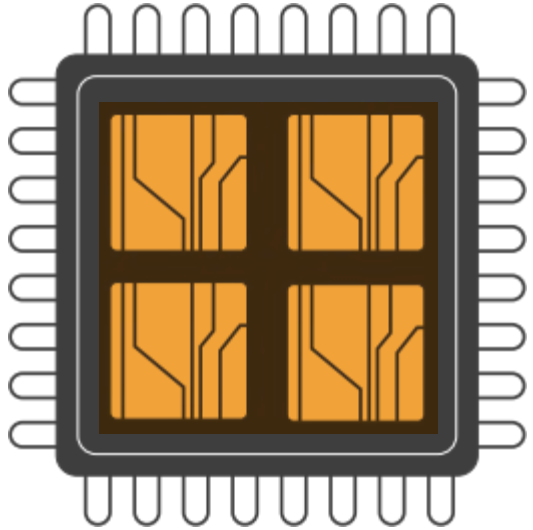


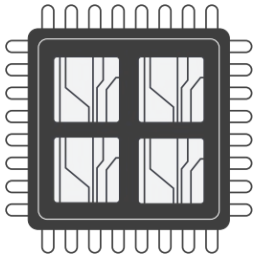





W
W
W
W

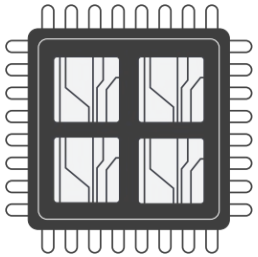






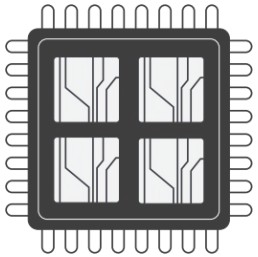
≃

ENERGIE **TRAVAIL**



≃

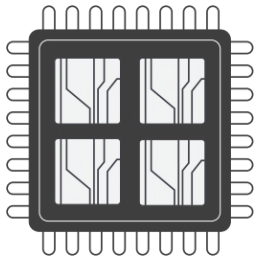
ENERGIE **TRAVAIL**

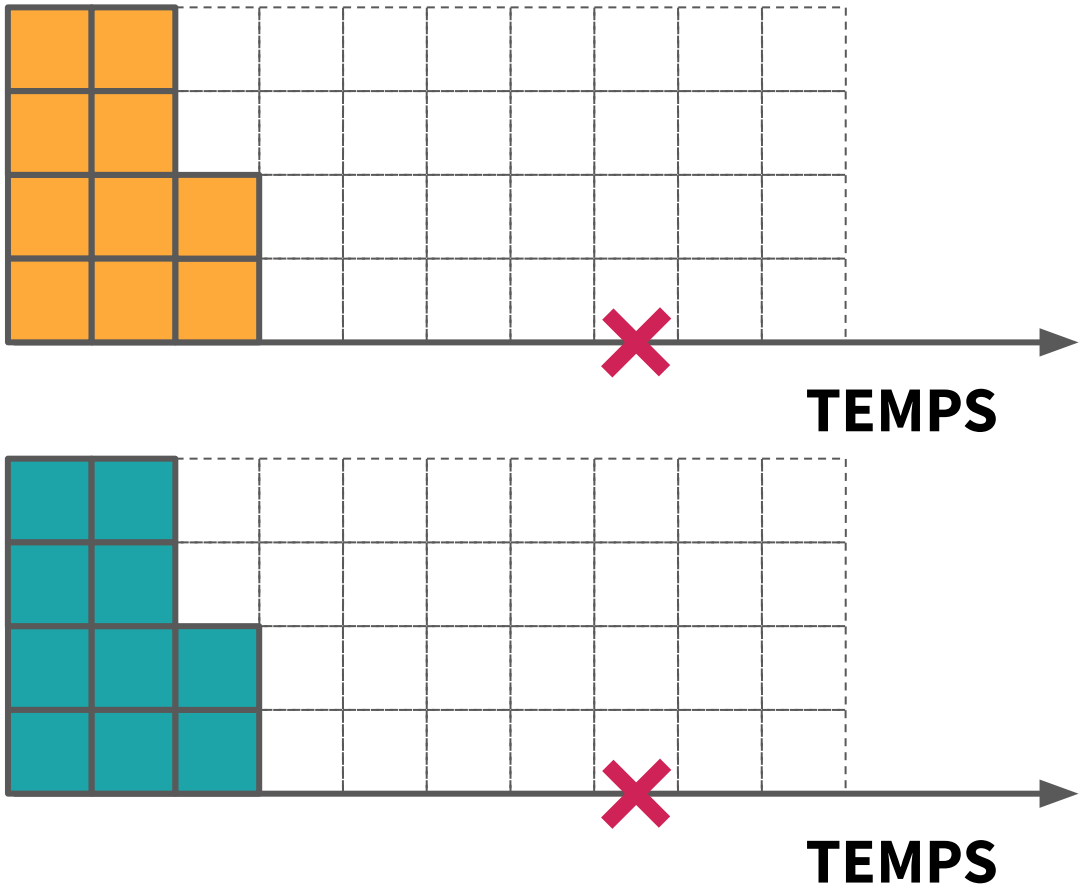


≃

ENERGIE **TRAVAIL**

Avantages du parallélisme

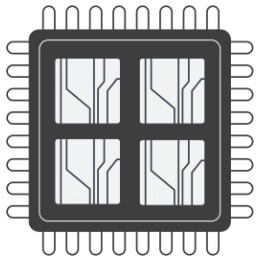




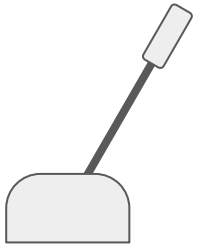
TRAVAIL 10

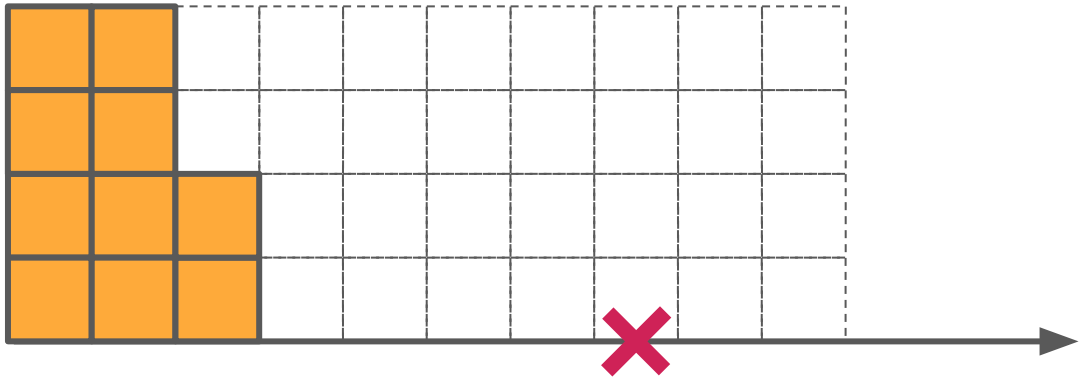
ENERGIE ~10

Avantages du parallélisme

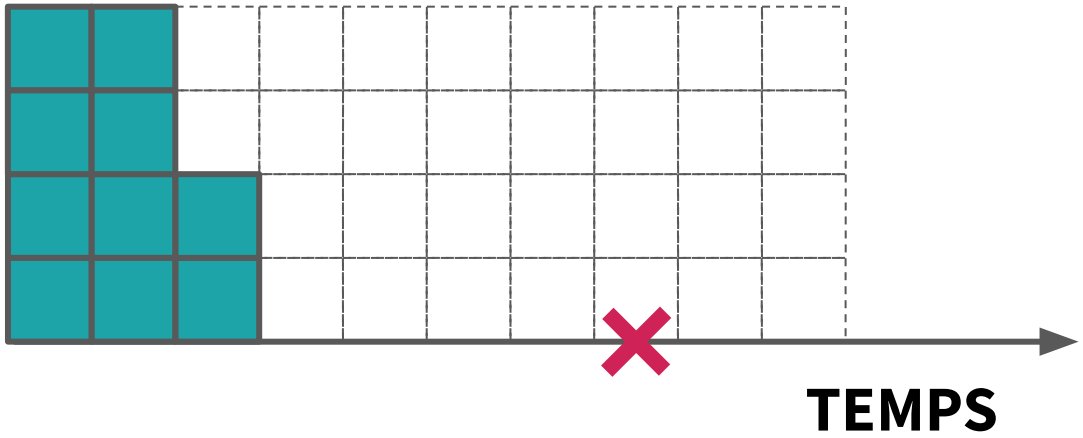


+



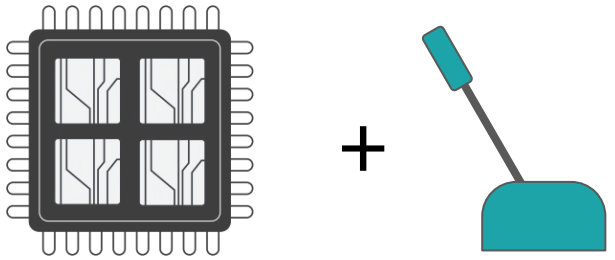


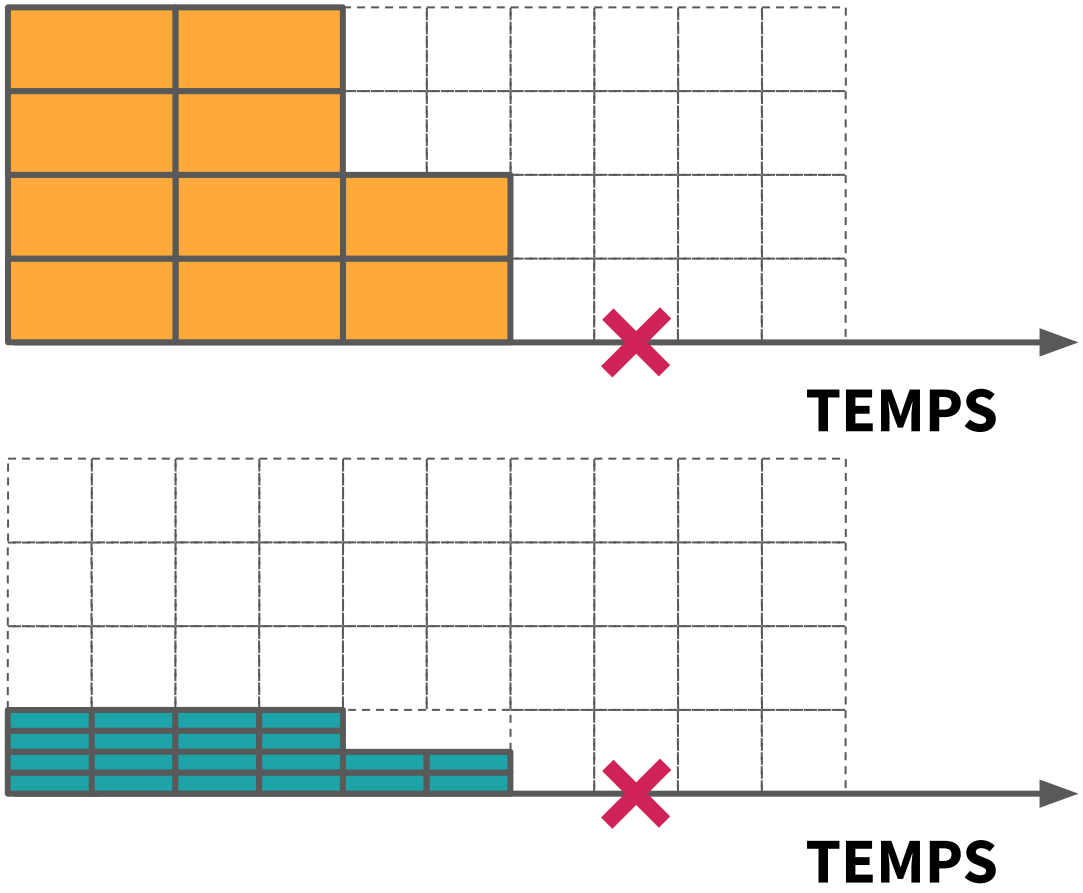
TRAVAIL 10



ENERGIE ~10

Avantages du parallélisme

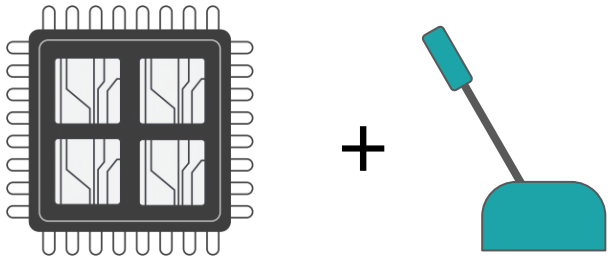


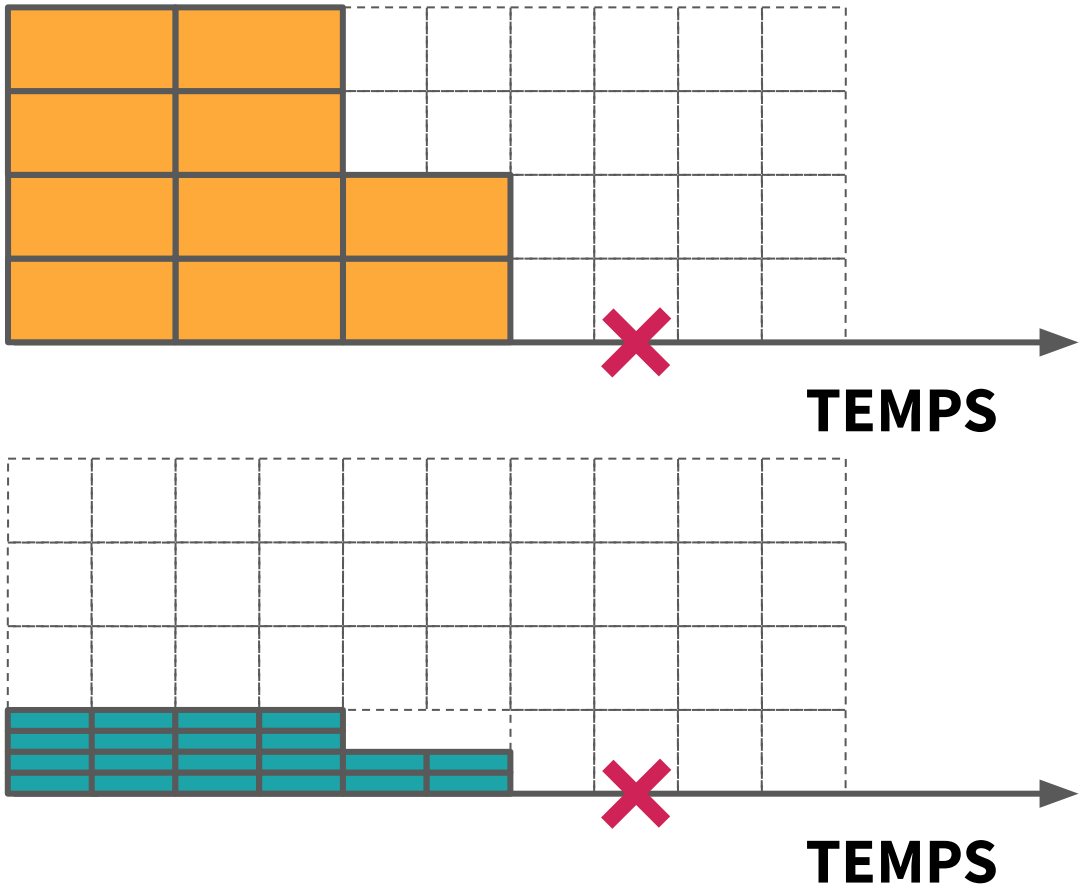


TRAVAIL 10

ENERGIE ~5

Avantages du parallélisme

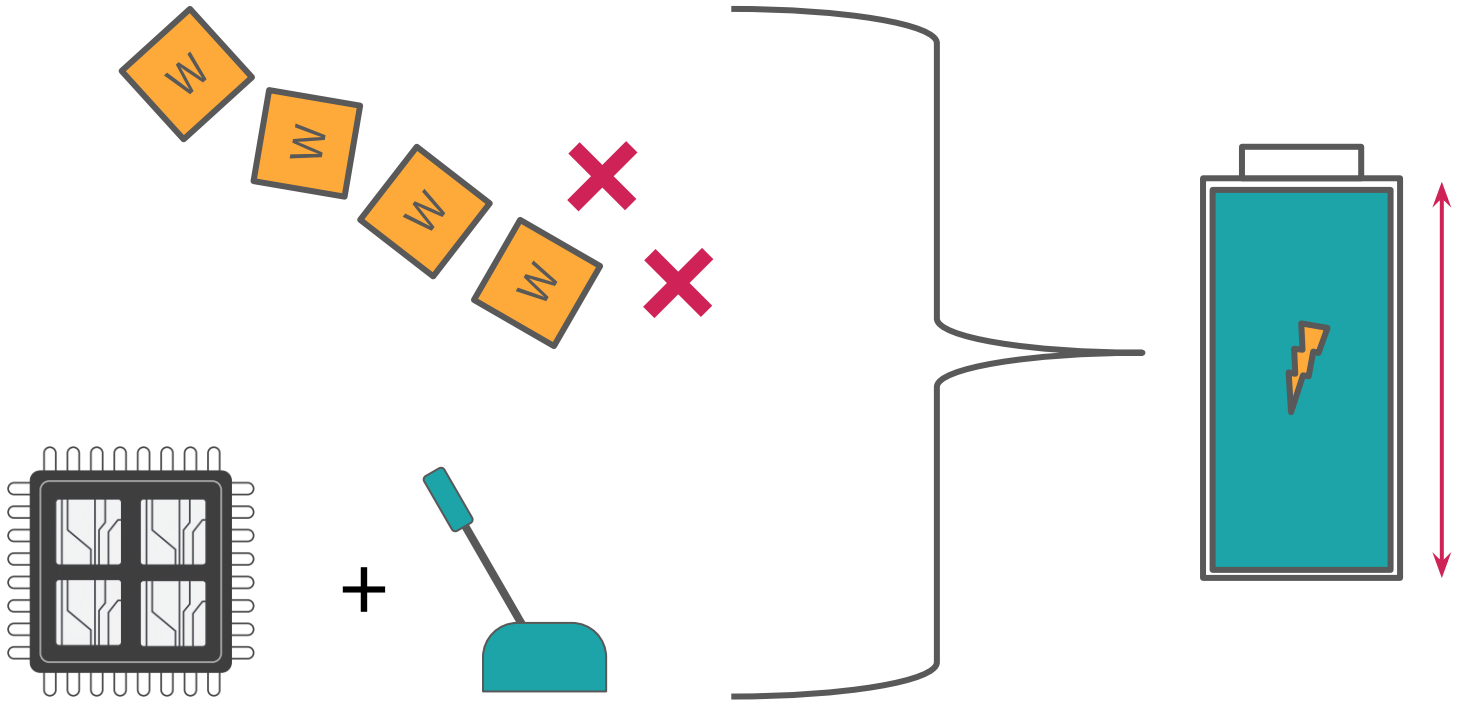


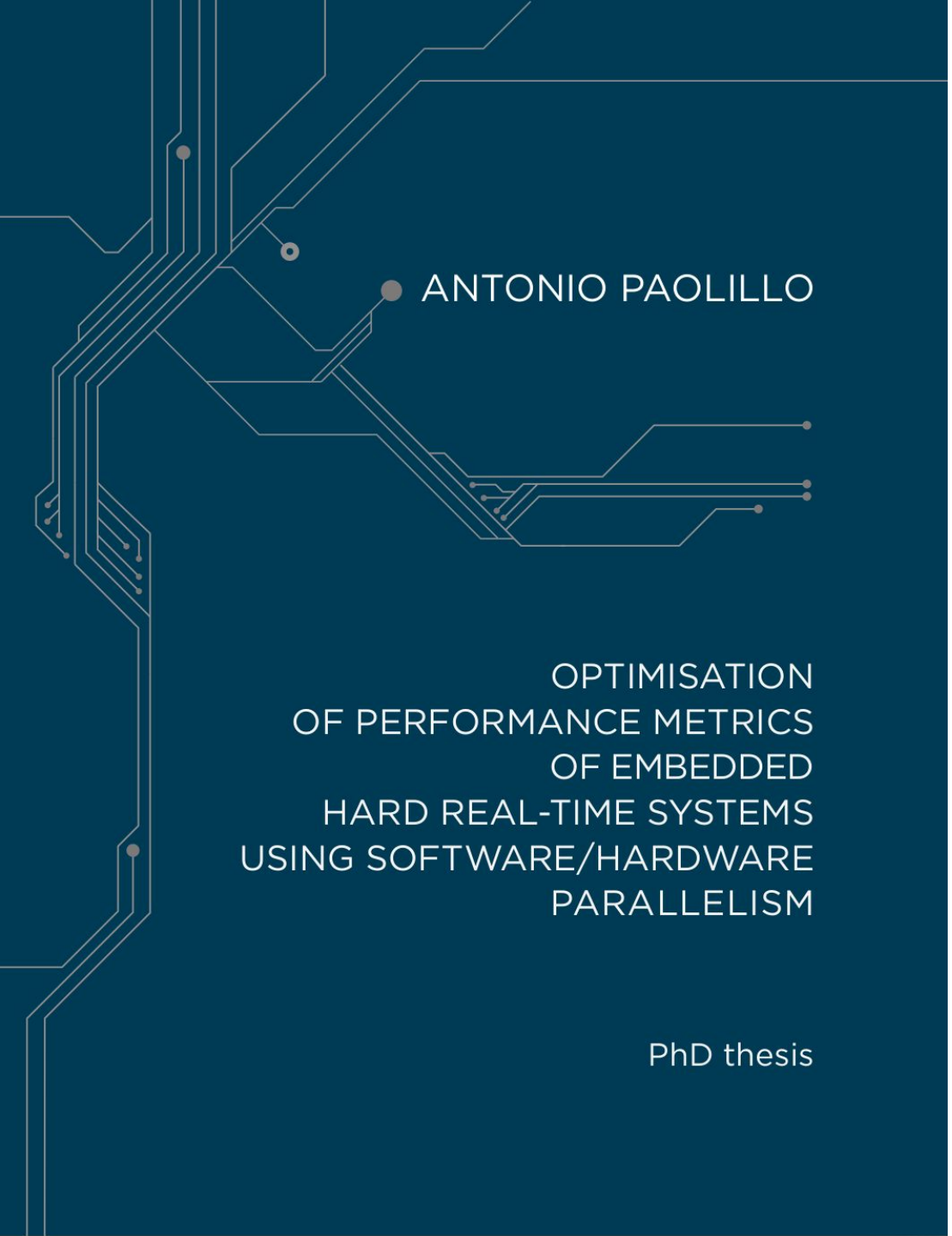


TRAVAIL 10

ENERGIE ~5 (-50%)

Le problème à résoudre

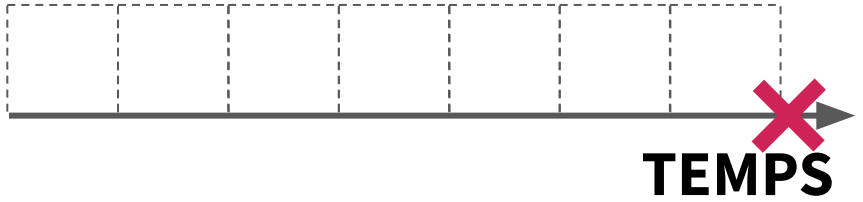



ANTONIO PAOLILLO
OPTIMISATION
OF PERFORMANCE METRICS
OF EMBEDDED
HARD REAL-TIME SYSTEMS
USING SOFTWARE/HARDWARE
PARALLELISM
PhD thesis

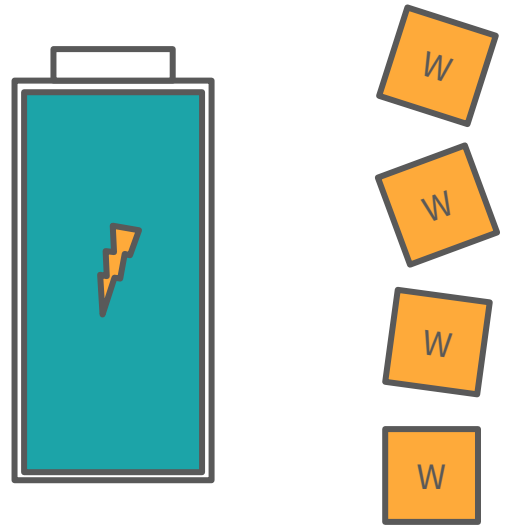
OPTIMISATION OF PERFORMANCE METRICS OF EMBEDDED HARD REAL-TIME SYSTEMS USING SOFTWARE/HARDWARE PARALLELISM

OPTIMISATION
OF PERFORMANCE METRICS
OF **EMBEDDED**
HARD REAL-TIME **SYSTEMS**
USING SOFTWARE/HARDWARE
PARALLELISM

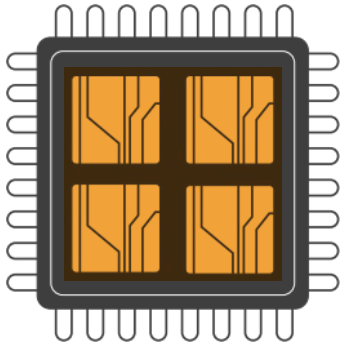
OPTIMISATION
OF PERFORMANCE METRICS
OF EMBEDDED
HARD REAL-TIME SYSTEMS
USING SOFTWARE/HARDWARE
PARALLELISM



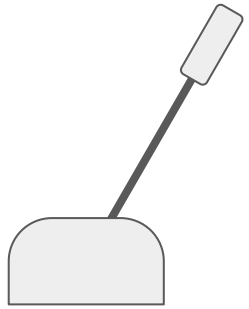
OPTIMISATION
OF **PERFORMANCE METRICS**
OF EMBEDDED
HARD REAL-TIME SYSTEMS
USING SOFTWARE/HARDWARE
PARALLELISM



OPTIMISATION
OF PERFORMANCE METRICS
OF EMBEDDED
HARD REAL-TIME SYSTEMS
USING SOFTWARE/**HARDWARE**
PARALLELISM



OPTIMISATION
OF PERFORMANCE METRICS
OF EMBEDDED
HARD REAL-TIME SYSTEMS
USING SOFTWARE/HARDWARE
PARALLELISM



-20%

-30%

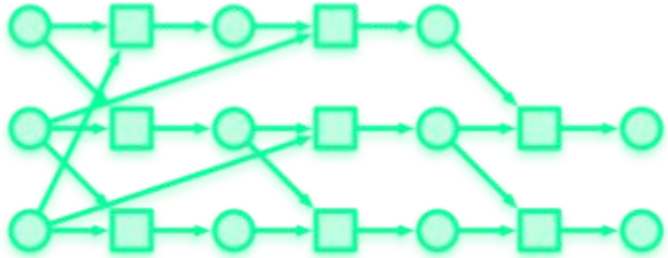
-50%

OPTIMISATION OF PERFORMANCE METRICS OF EMBEDDED HARD REAL-TIME SYSTEMS USING SOFTWARE/HARDWARE PARALLELISM

OPTIMISATION
OF PERFORMANCE METRICS
OF EMBEDDED
HARD REAL-TIME SYSTEMS
USING SOFTWARE/HARDWARE
PARALLELISM

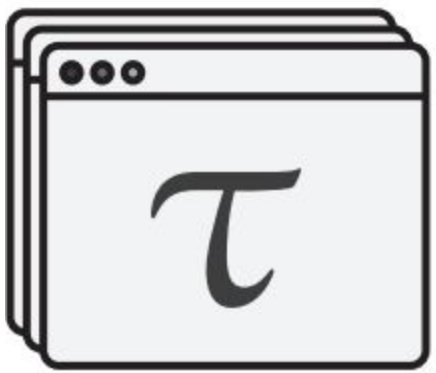
?

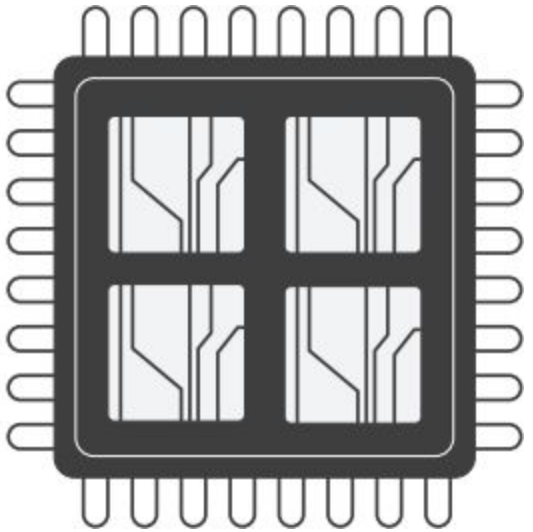
OPTIMISATION
OF PERFORMANCE METRICS
OF EMBEDDED
HARD REAL-TIME SYSTEMS
USING **SOFTWARE**/HARDWARE
PARALLELISM

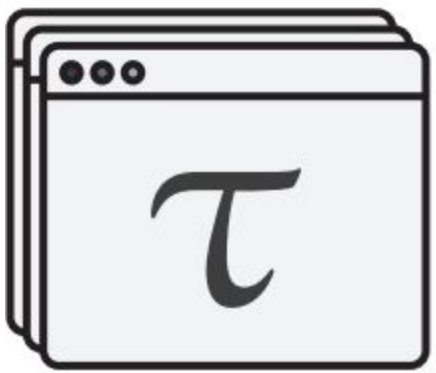


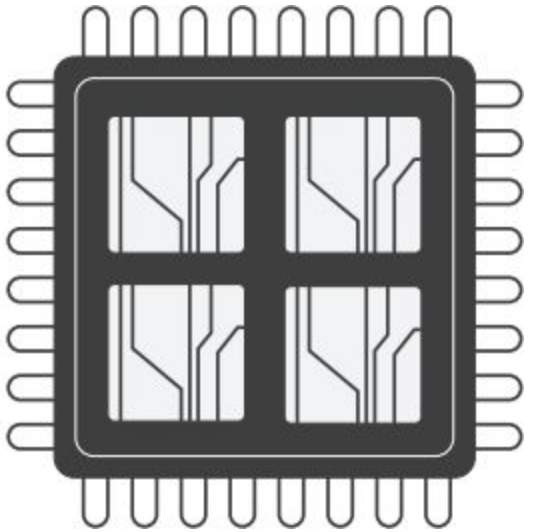
Agenda

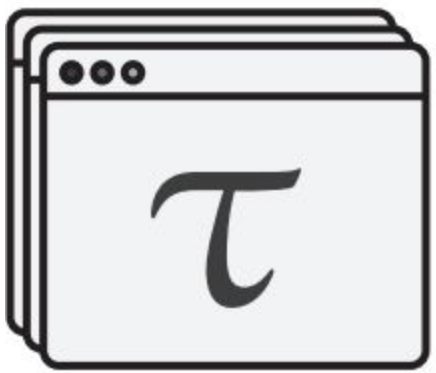
1. Domaine, problème, intuition
2. **Contributions et thèse**
3. Directions futures

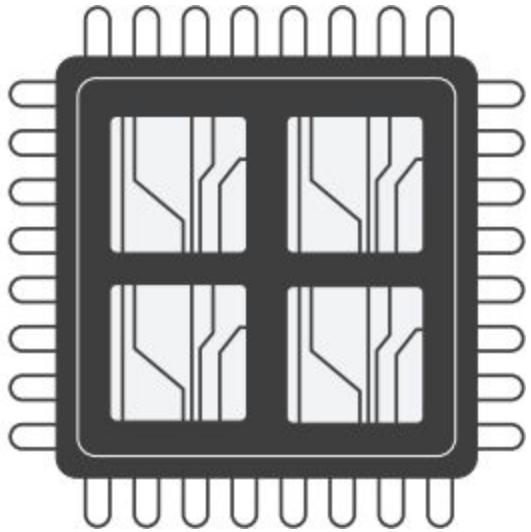

τ

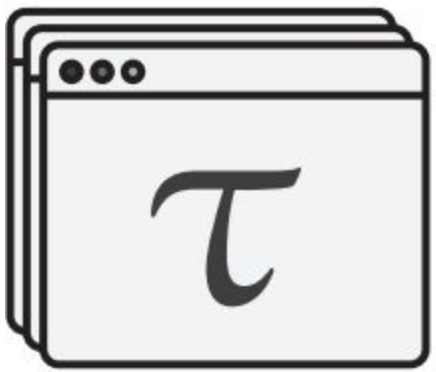


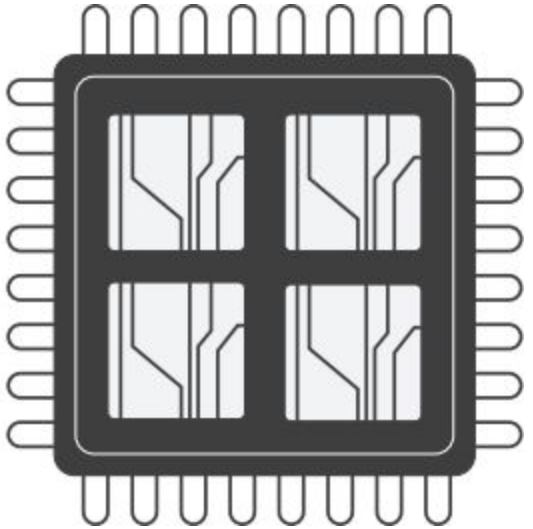

τ

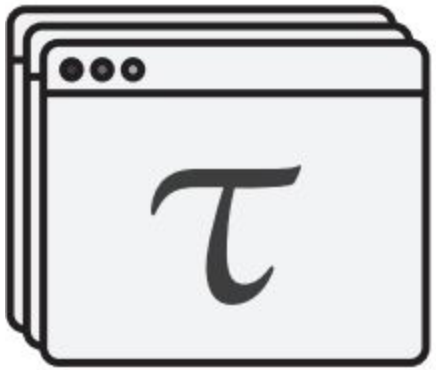


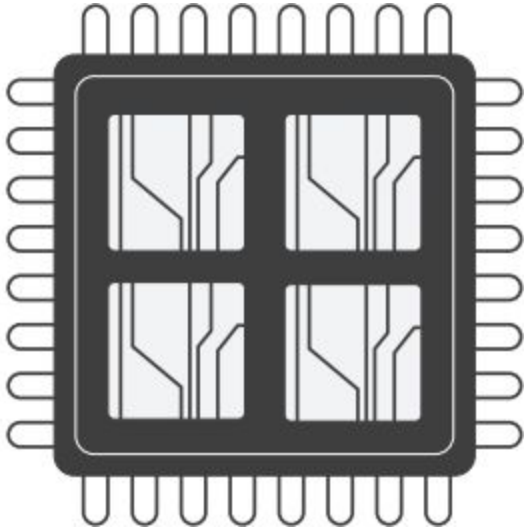

τ

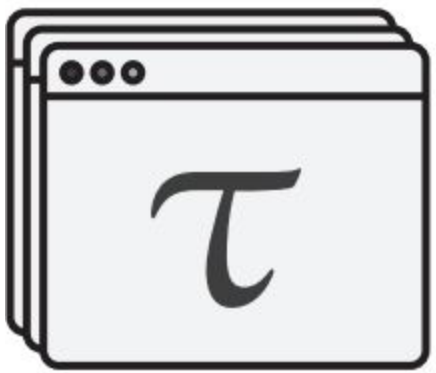


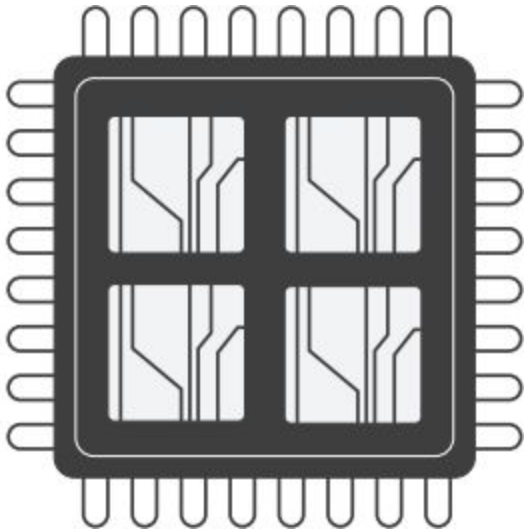

τ

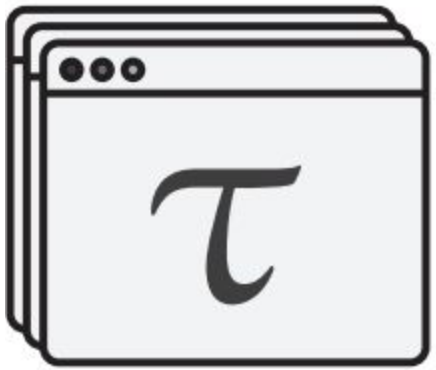


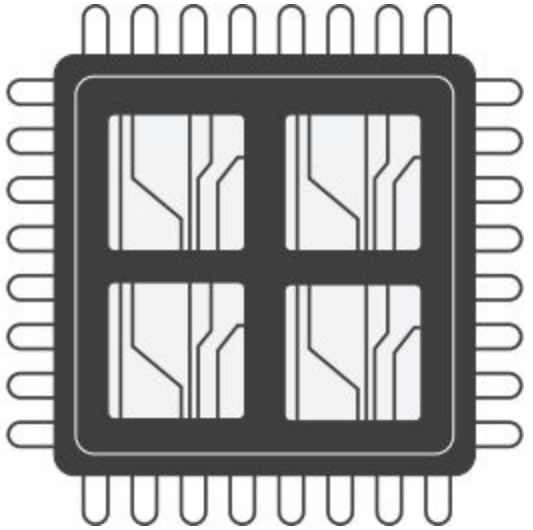

τ

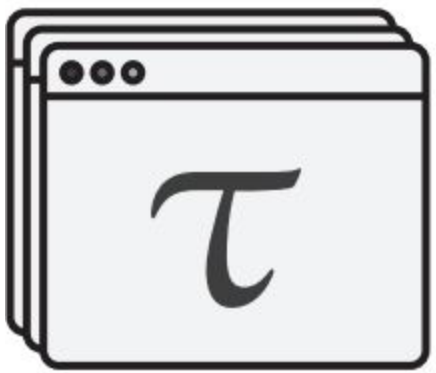


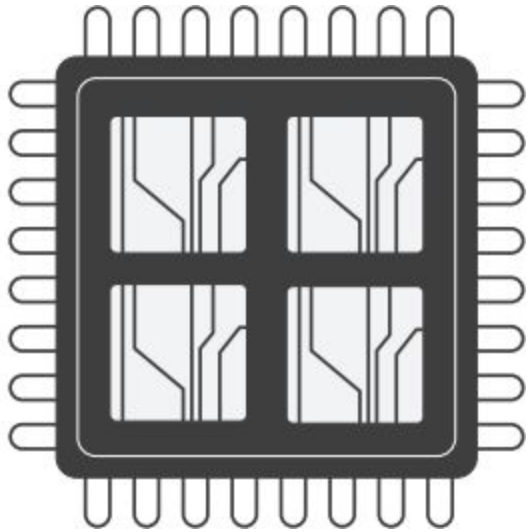

τ

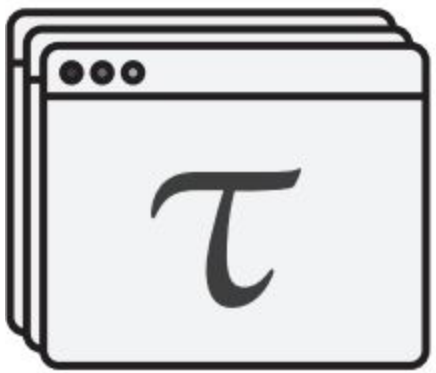


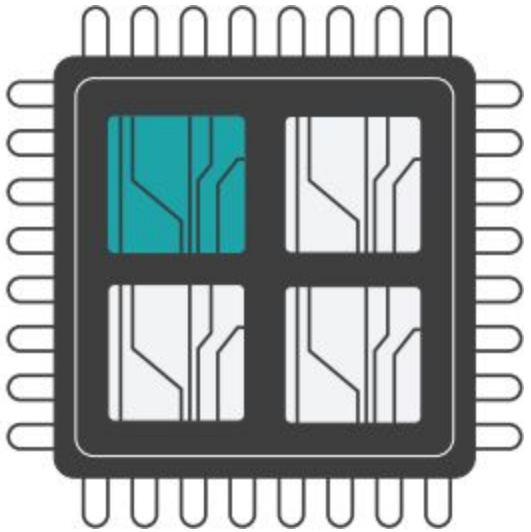

τ

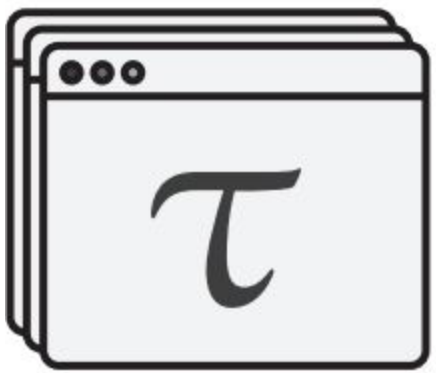



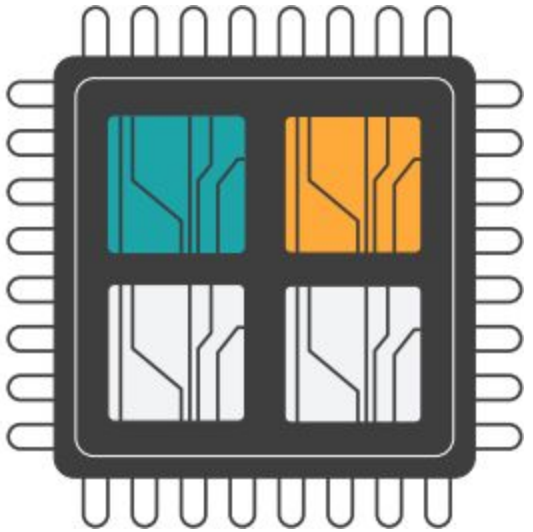

τ

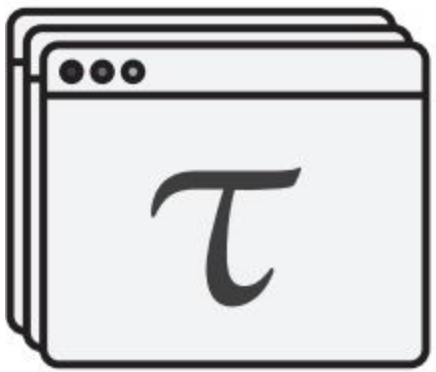



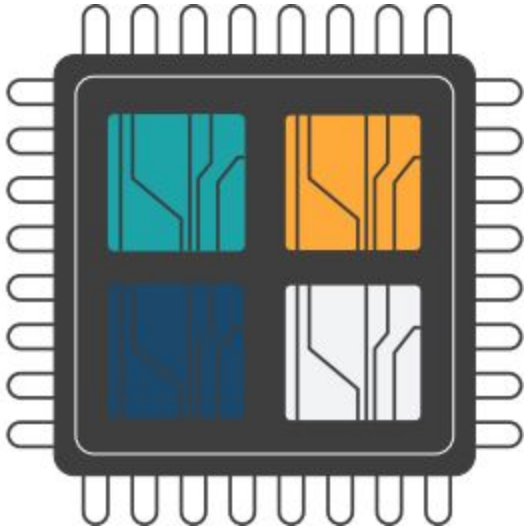

τ

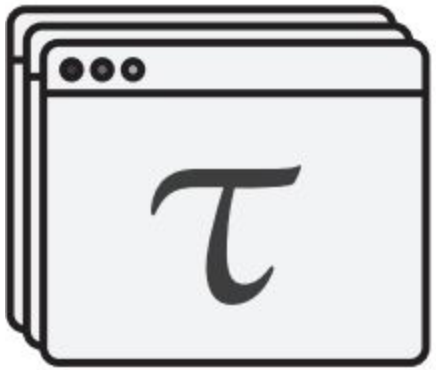



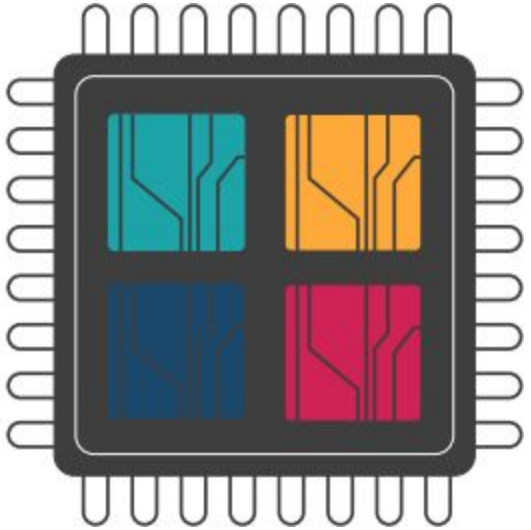

τ

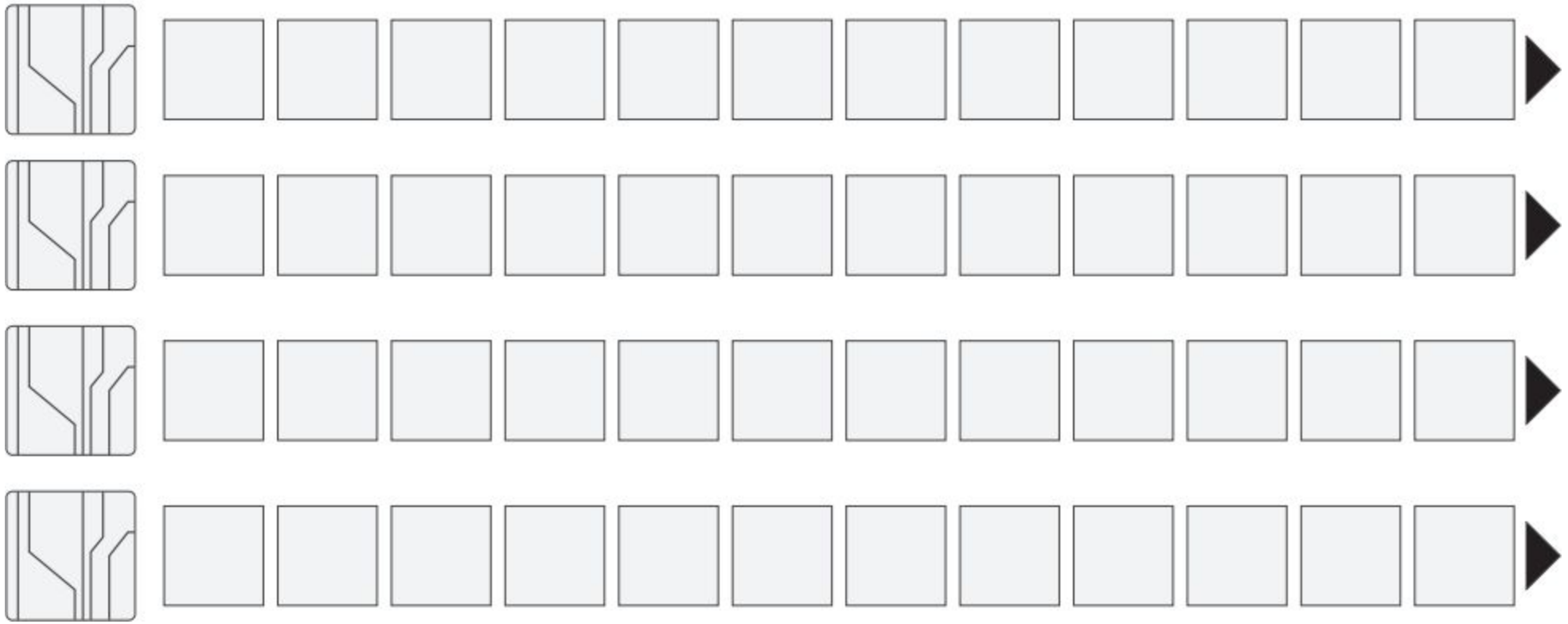

τ

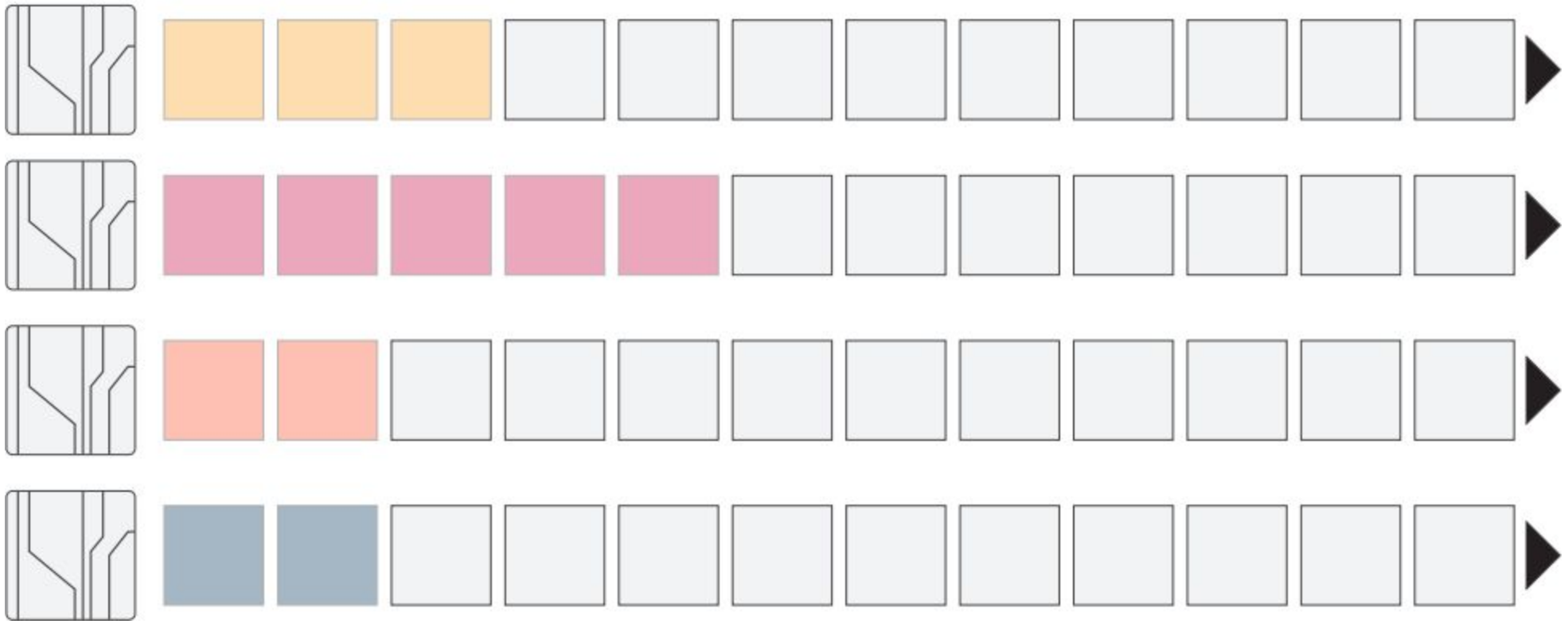

τ

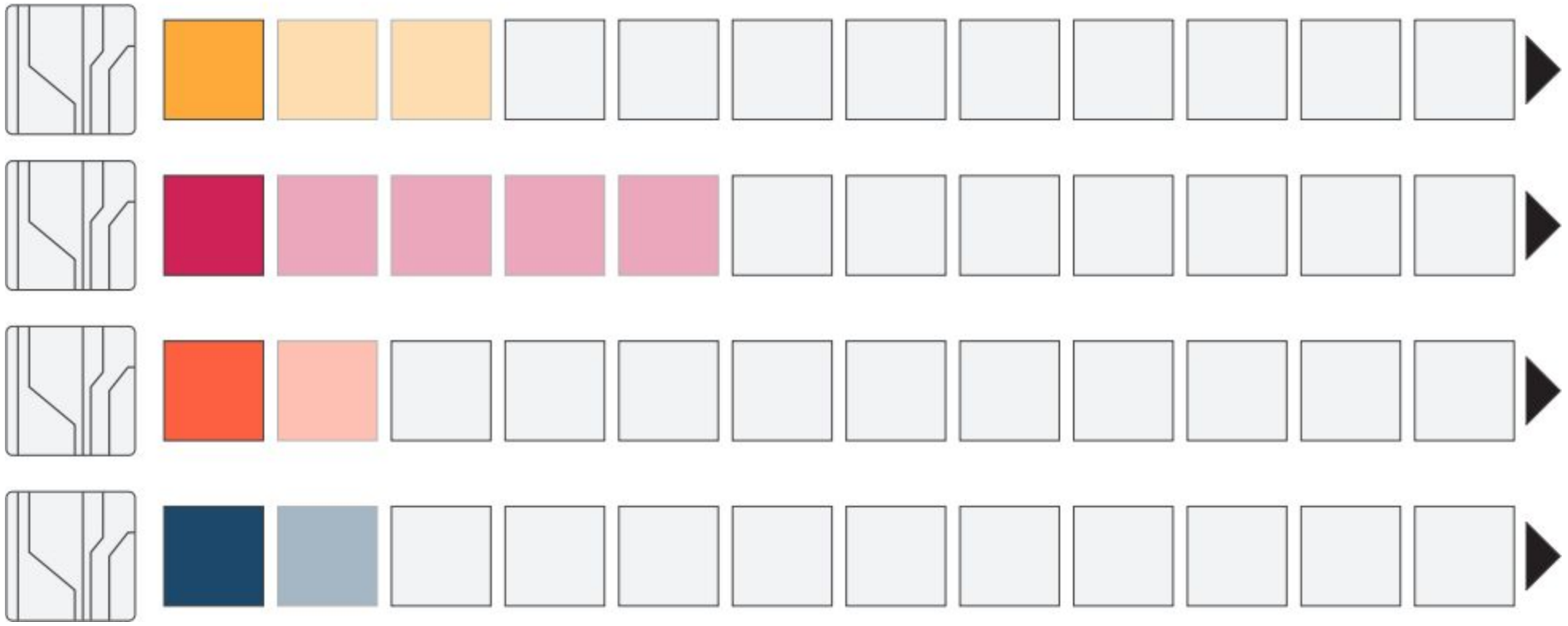
Ordonnancement séquentiel



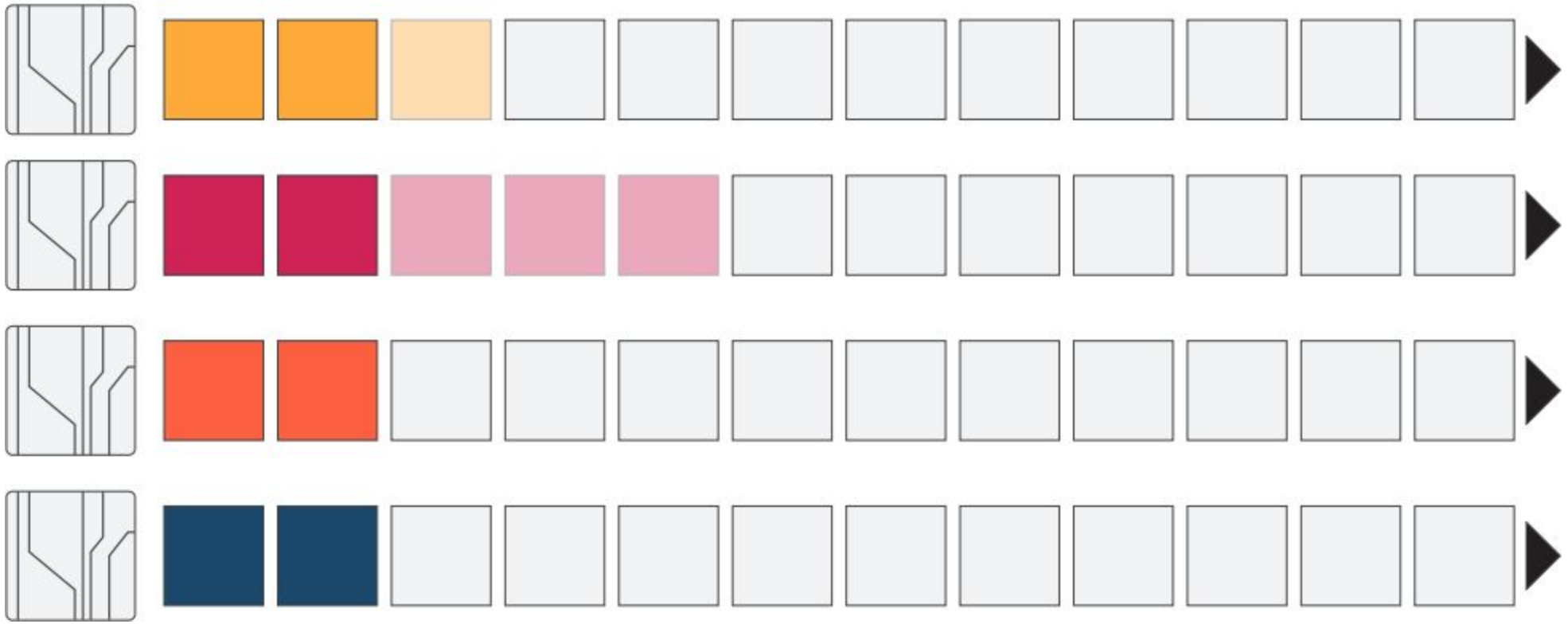
Ordonnancement séquentiel



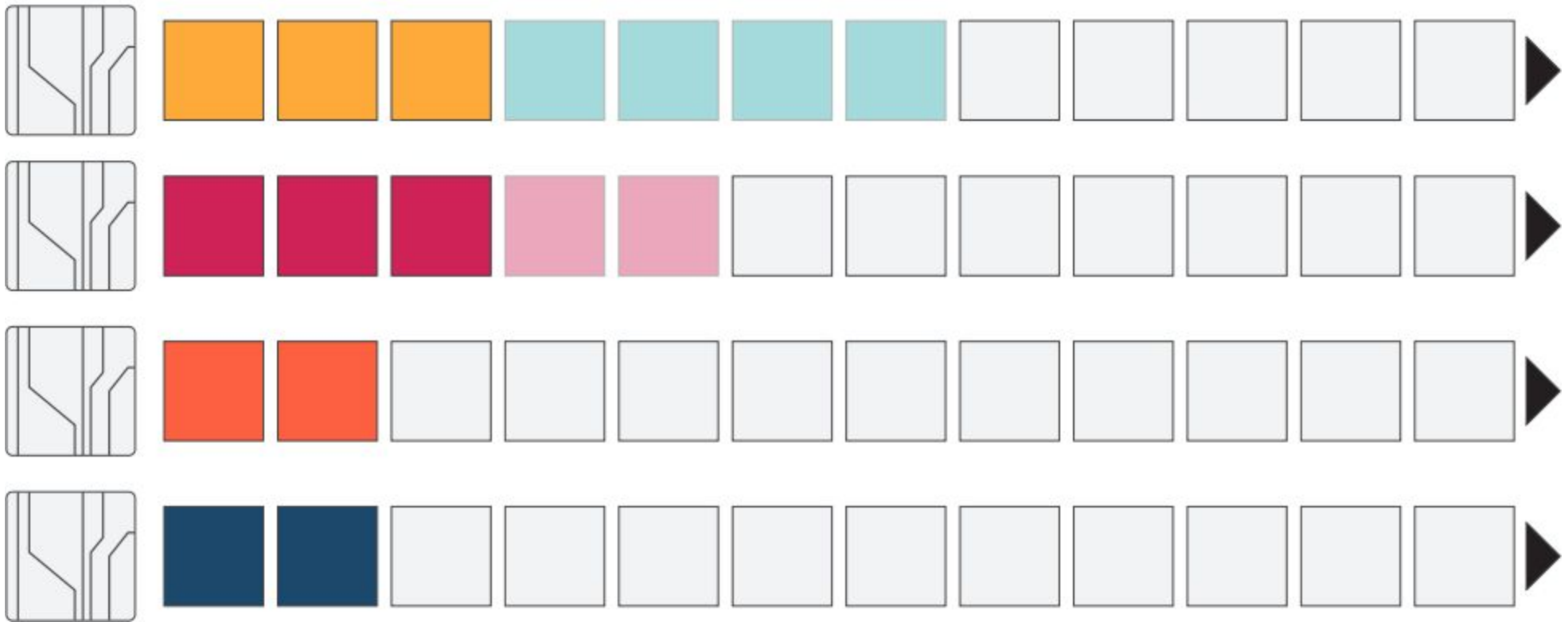
Ordonnancement séquentiel



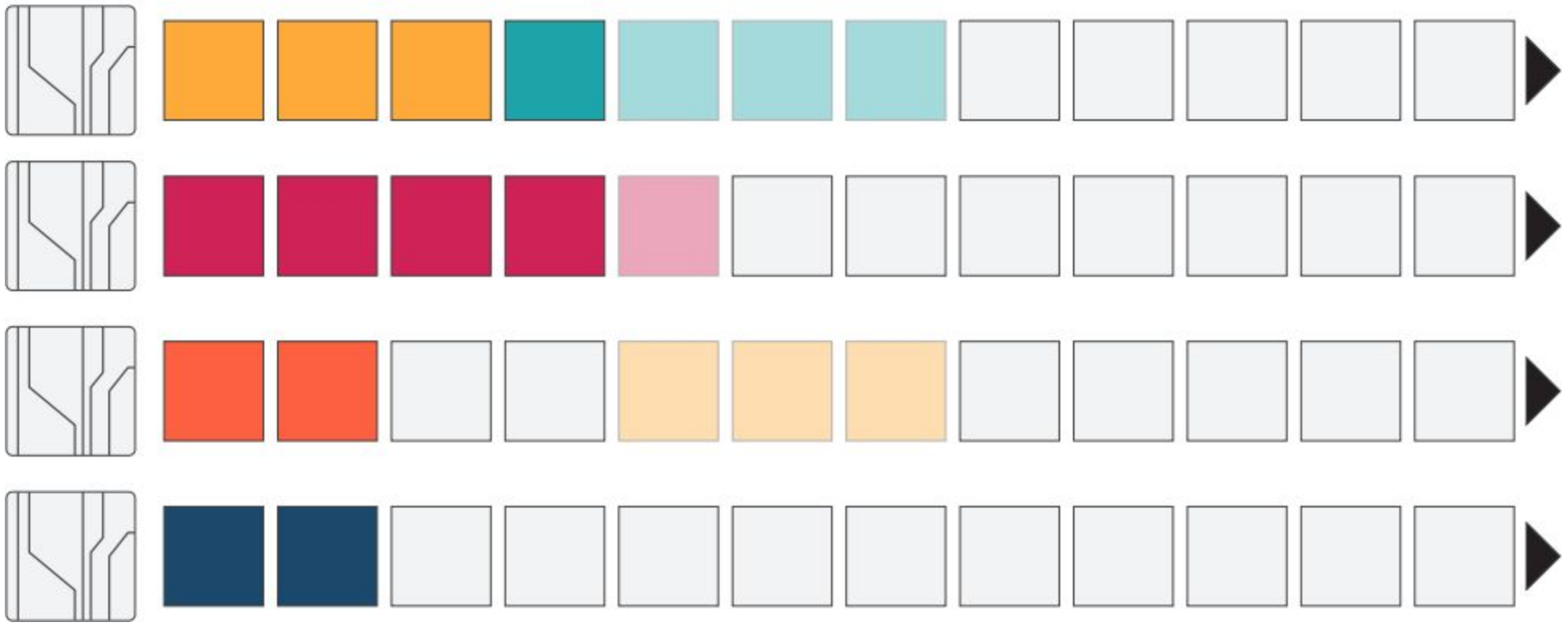
Ordonnancement séquentiel



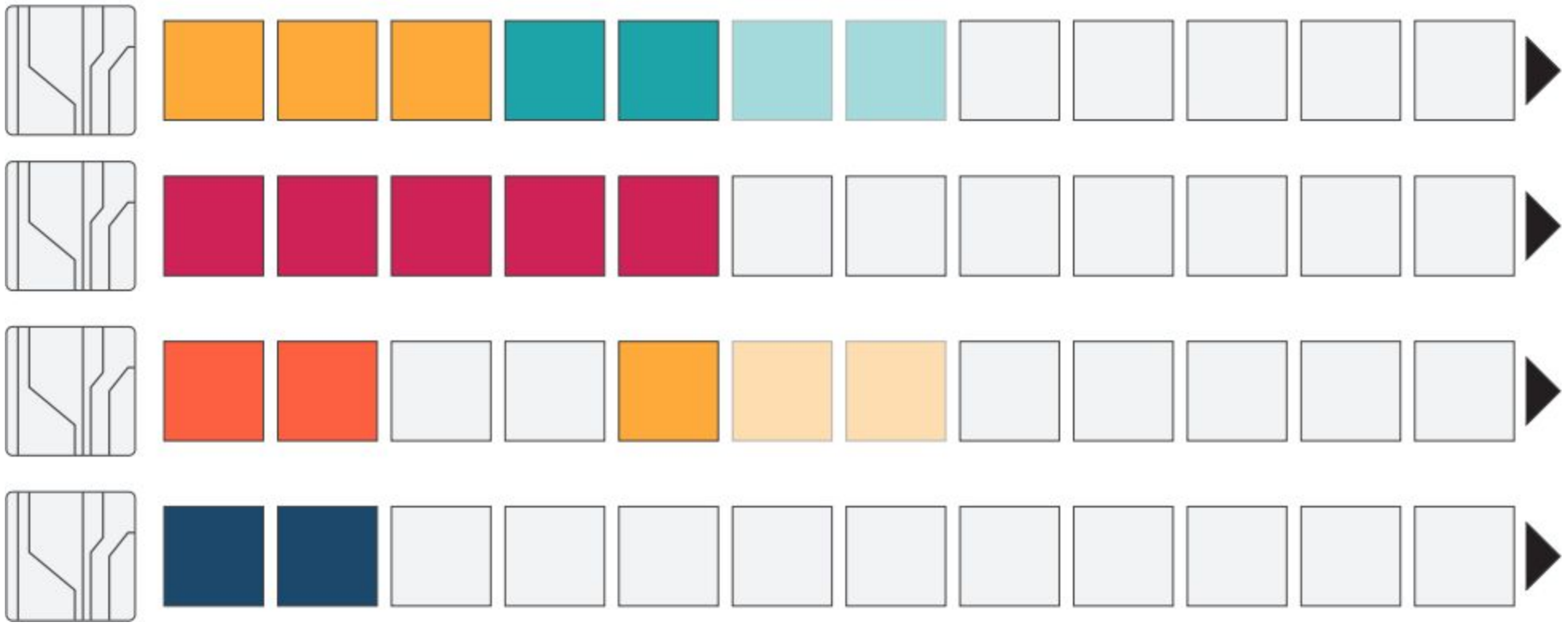
Ordonnancement séquentiel



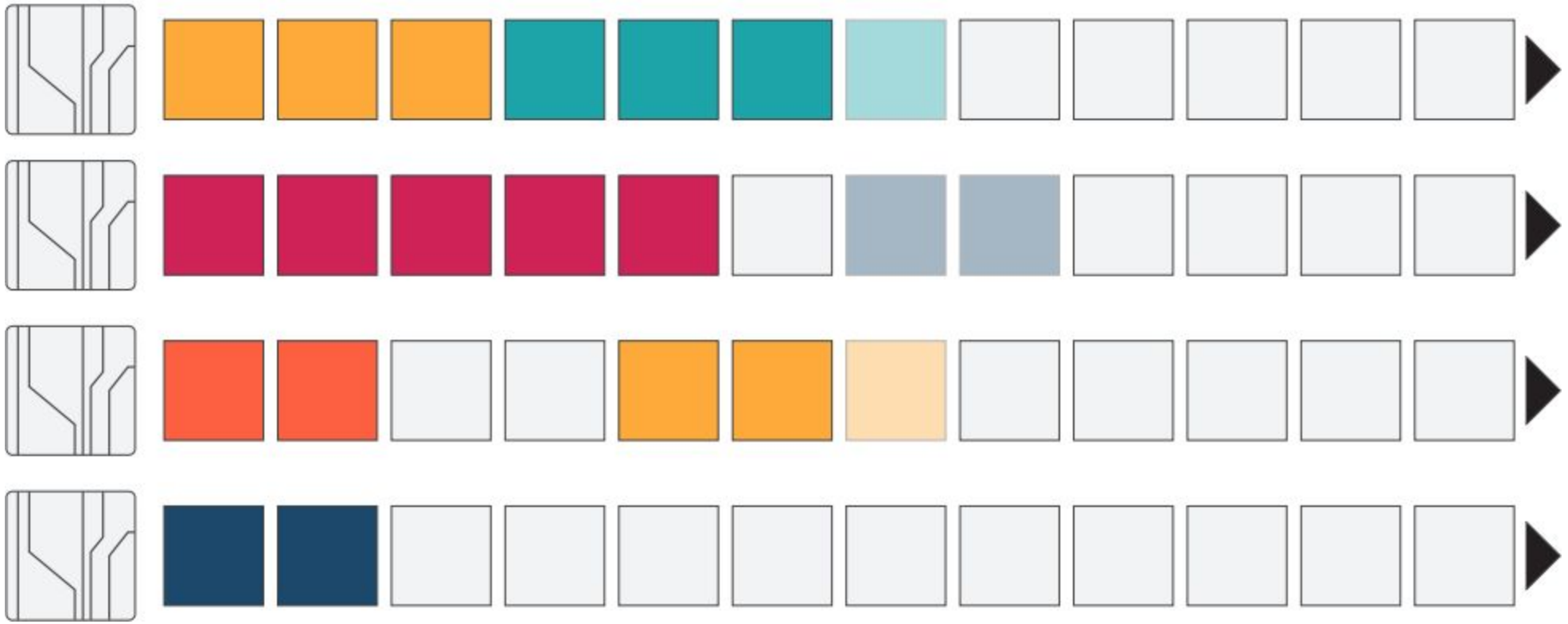
Ordonnancement séquentiel



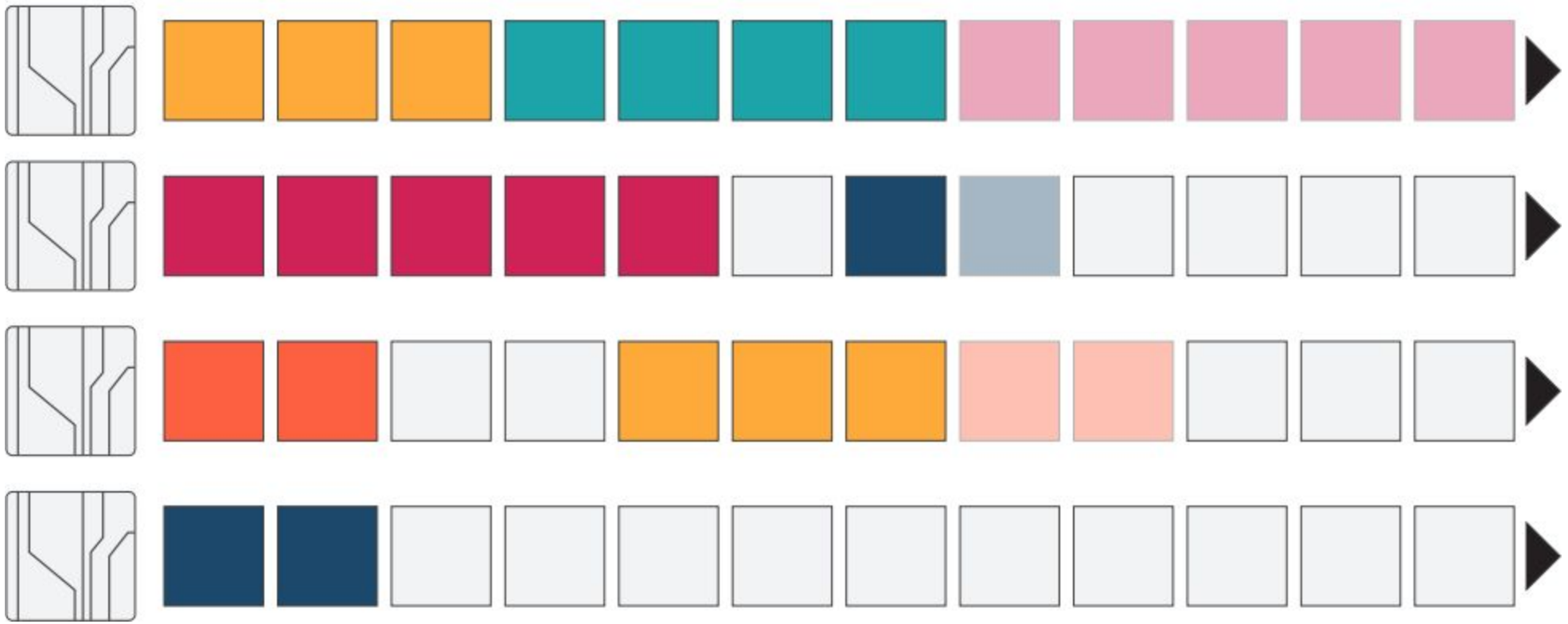
Ordonnancement séquentiel



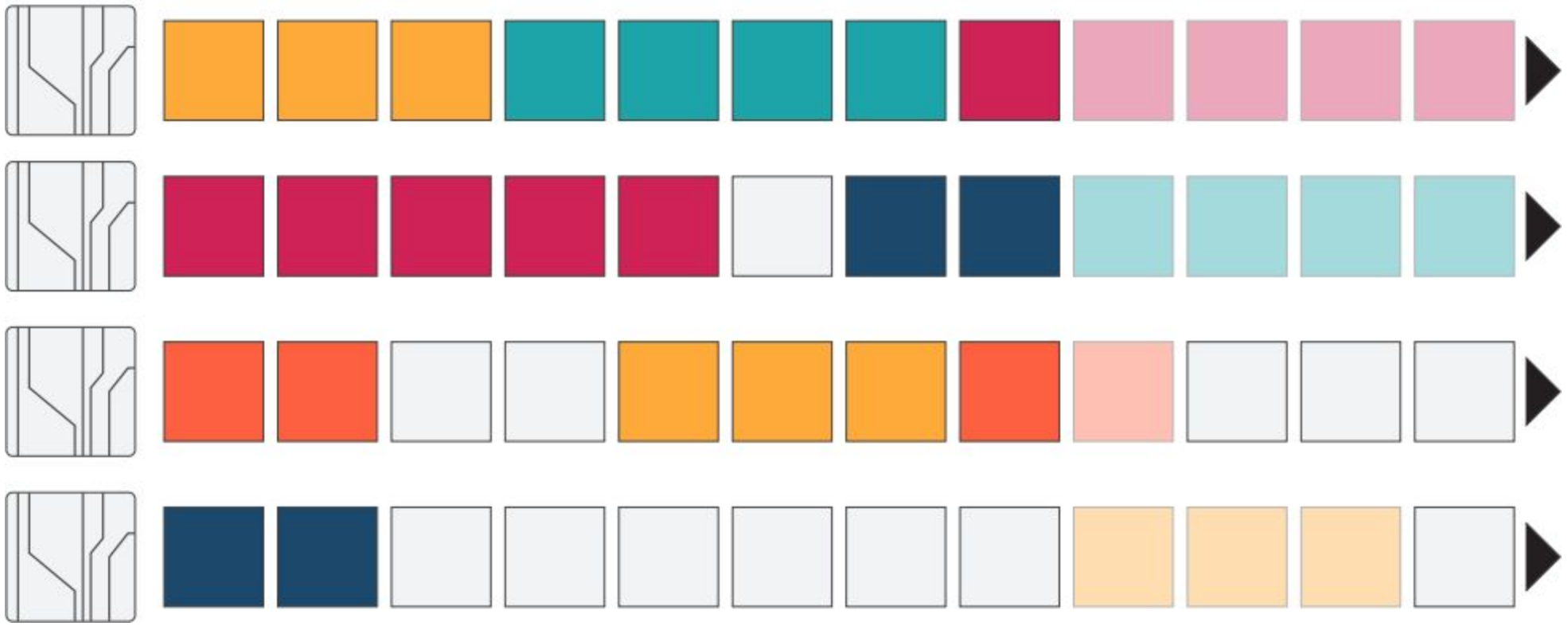
Ordonnancement séquentiel



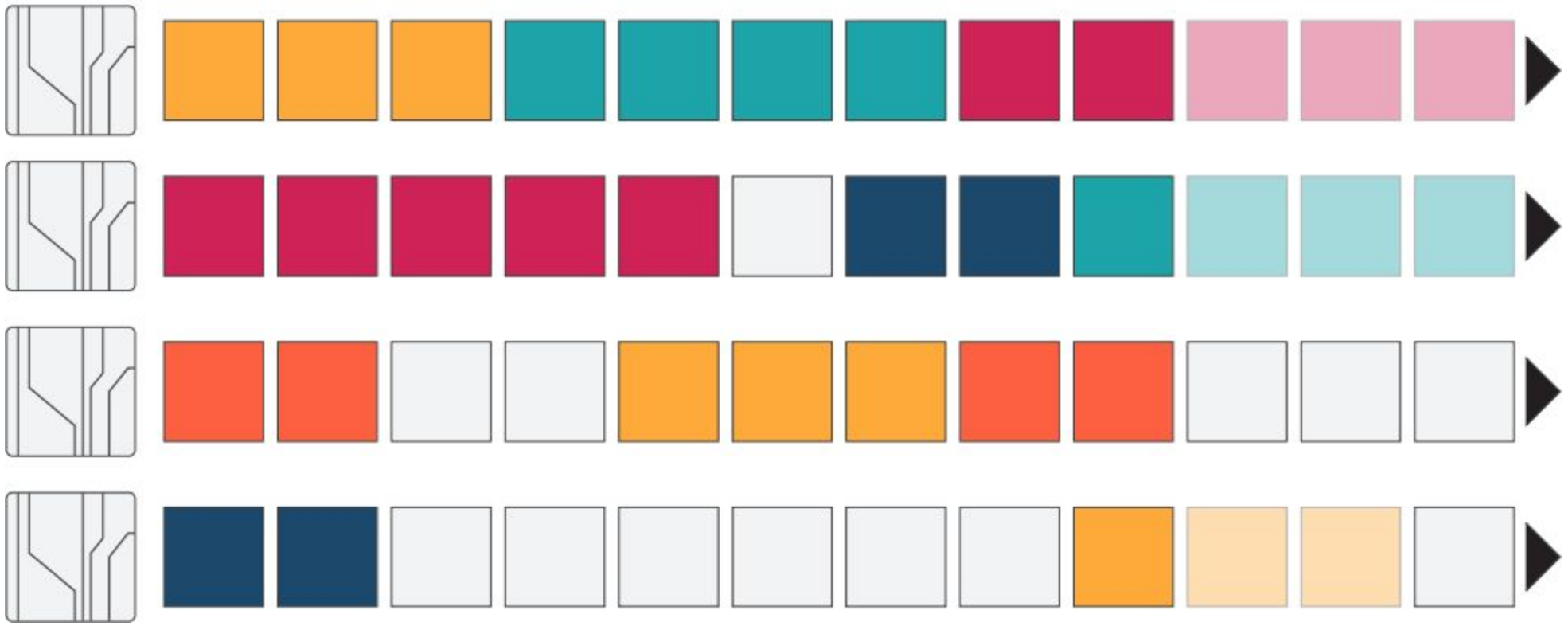
Ordonnancement séquentiel



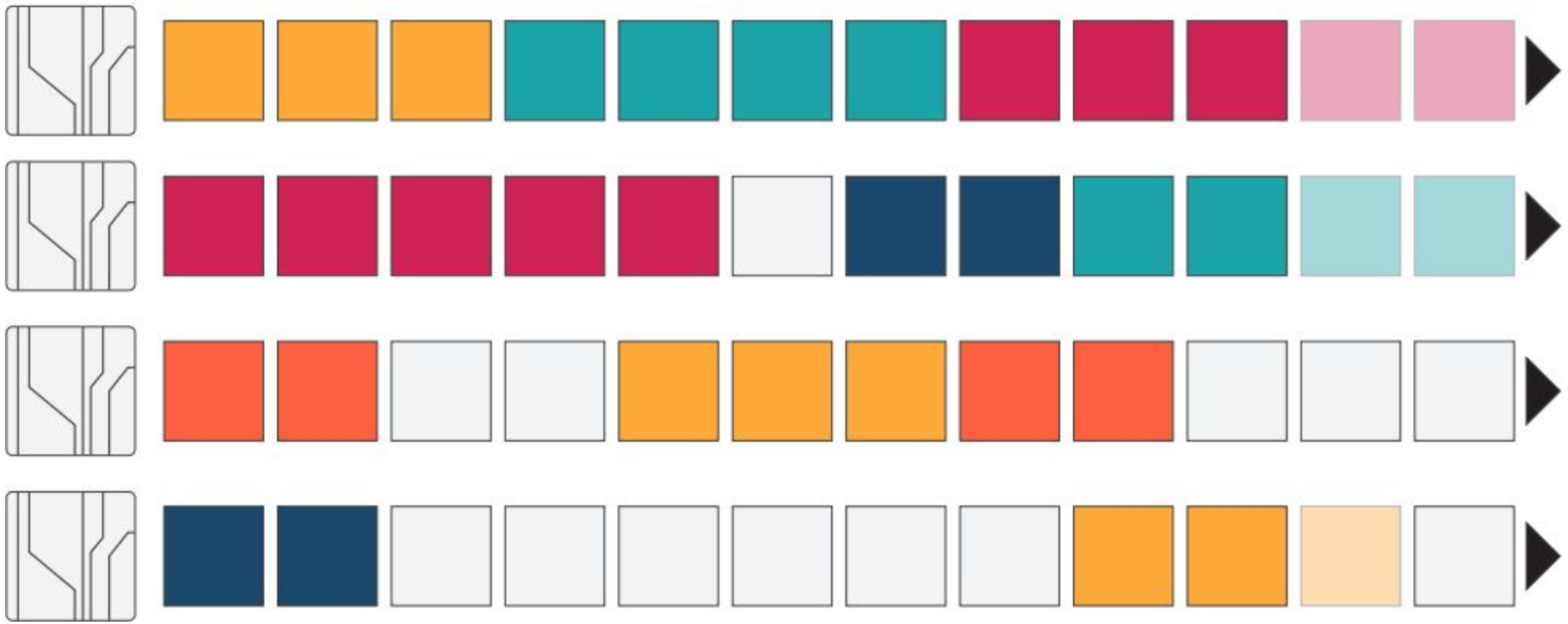
Ordonnancement séquentiel



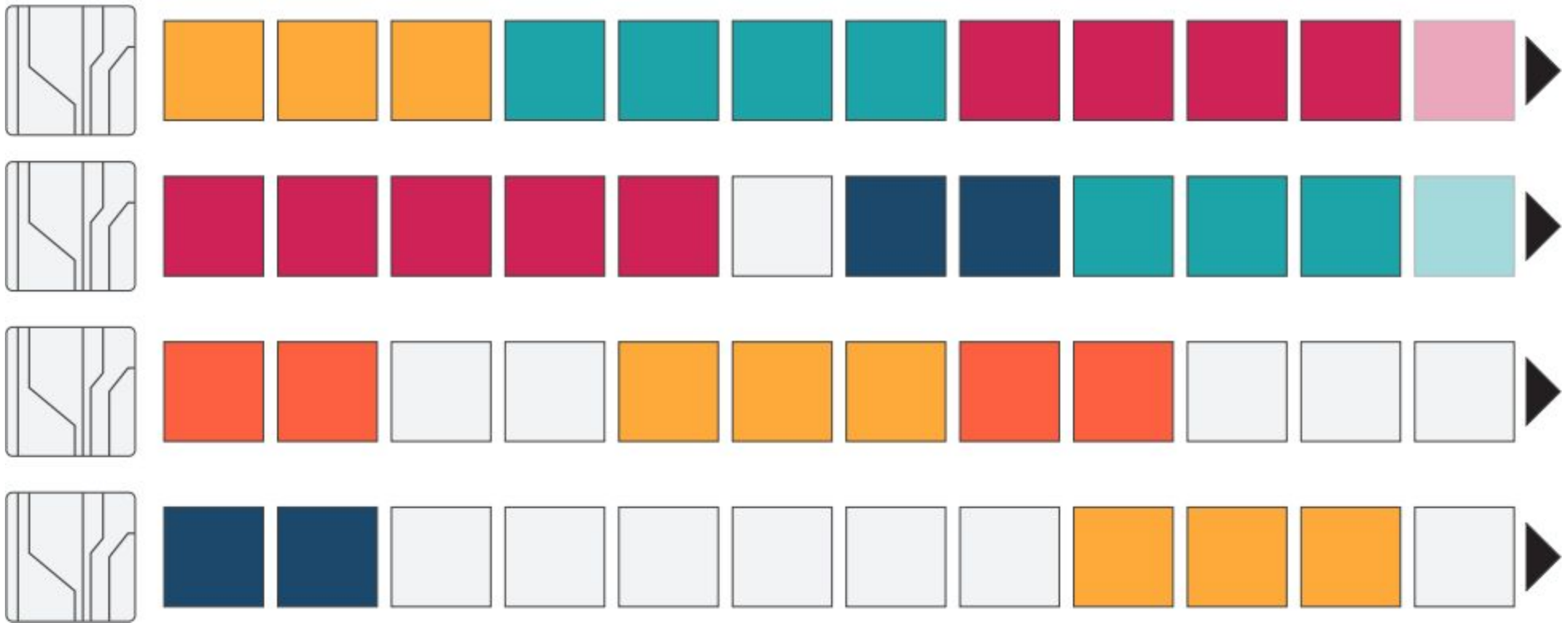
Ordonnancement séquentiel



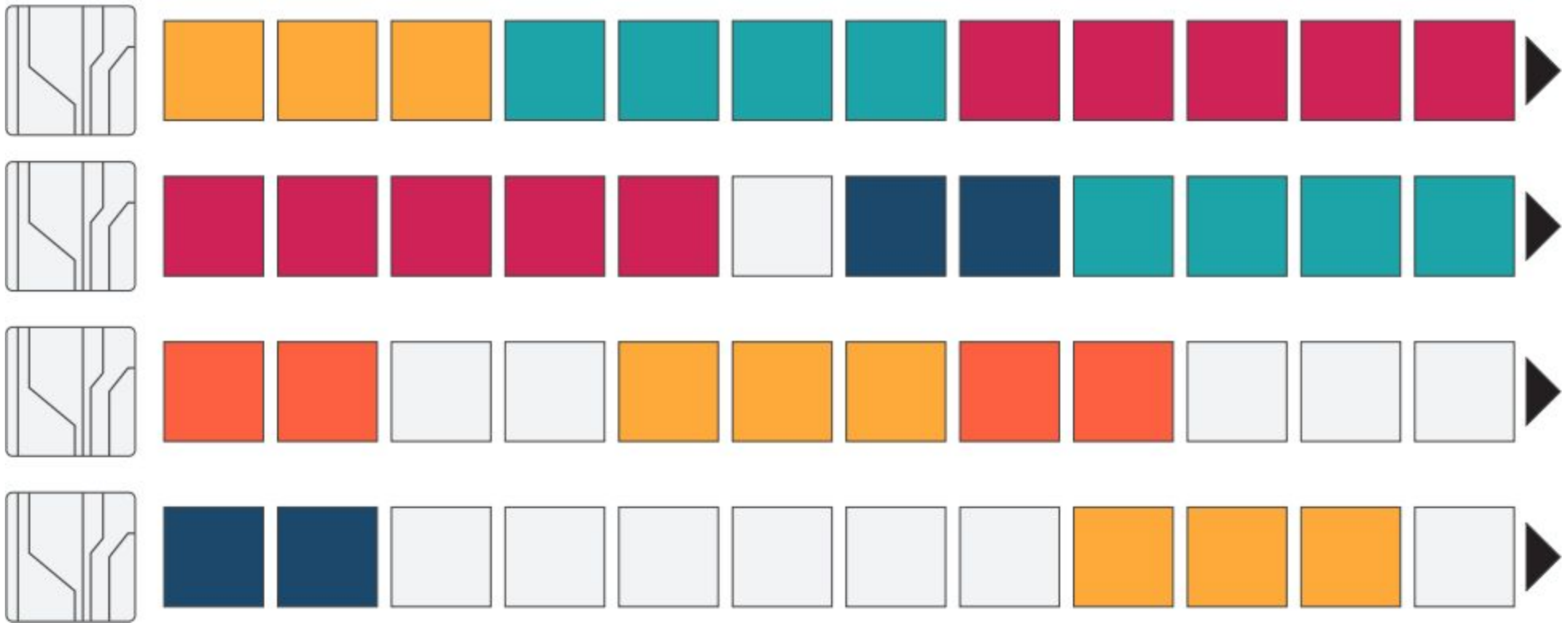
Ordonnancement séquentiel



Ordonnancement séquentiel



Ordonnancement séquentiel



But

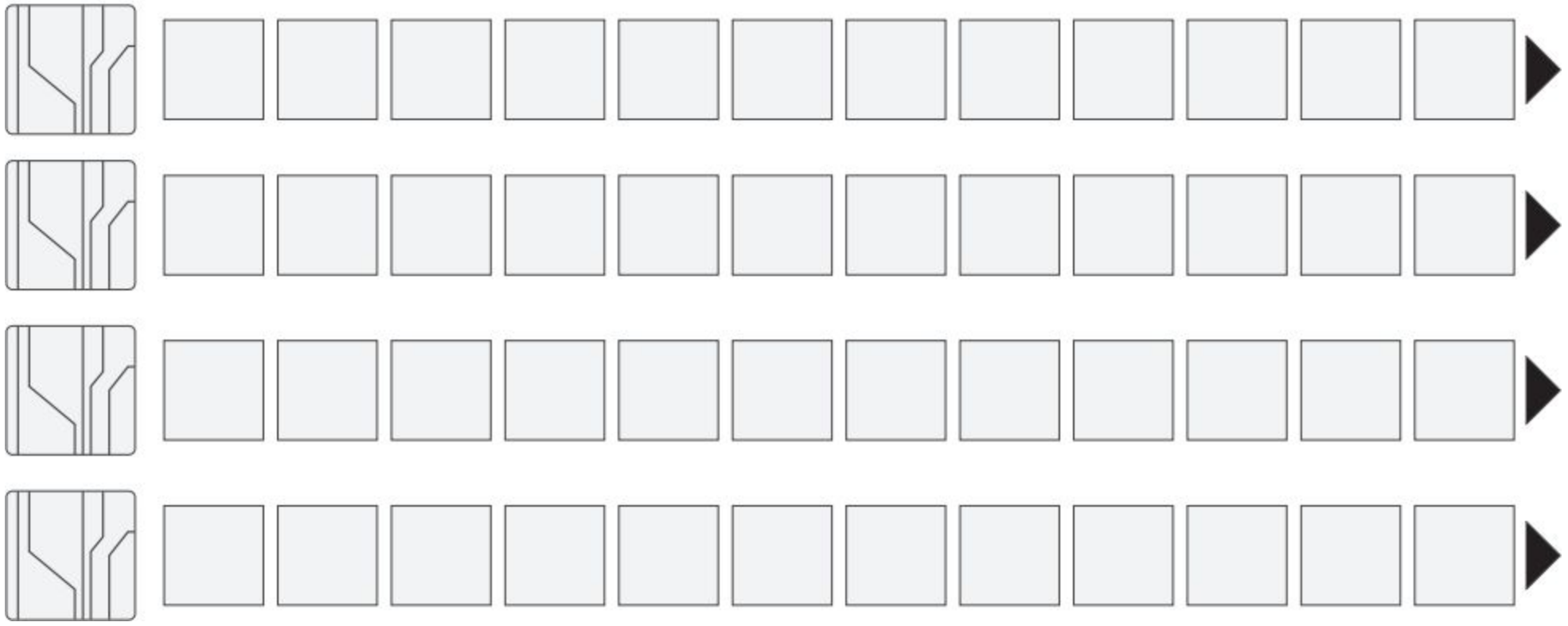
=

réduire l’énergie consommée

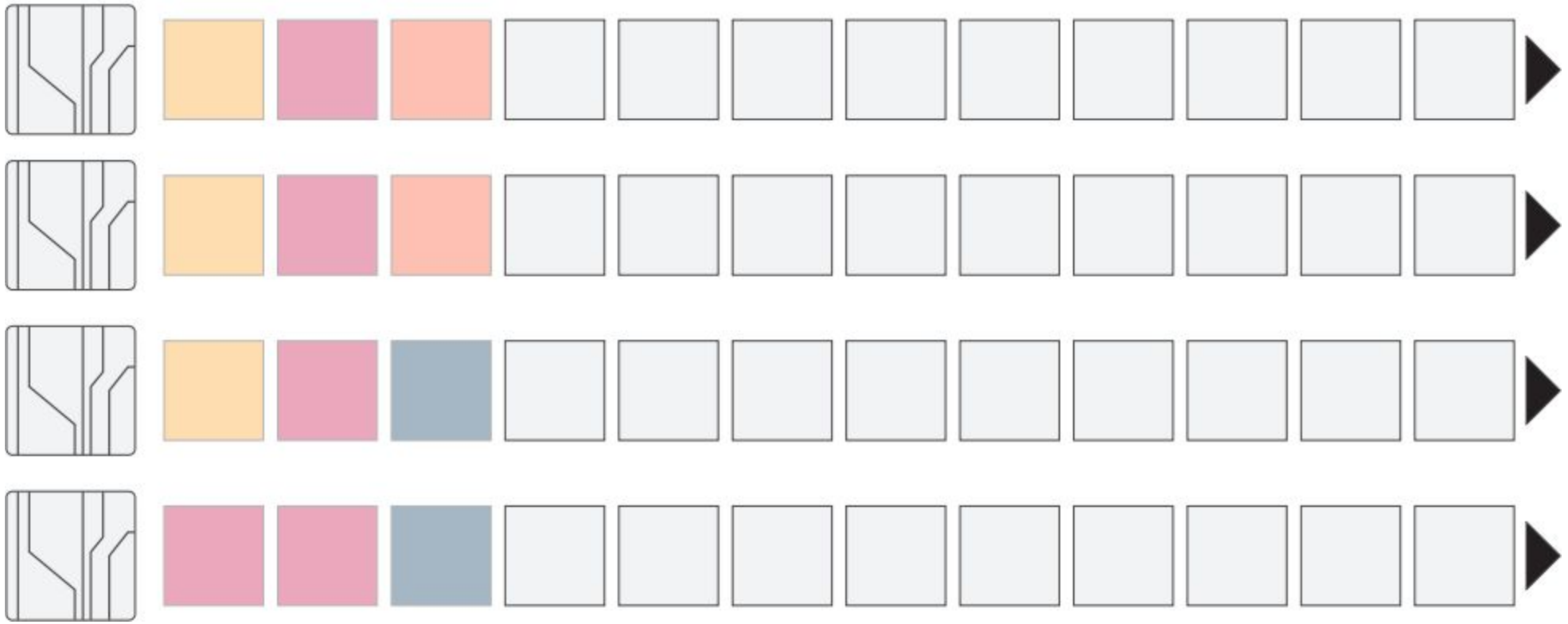
"The minimum **speed** is limited by the **sequential** job model"

- J. Anderson, S. Baruah

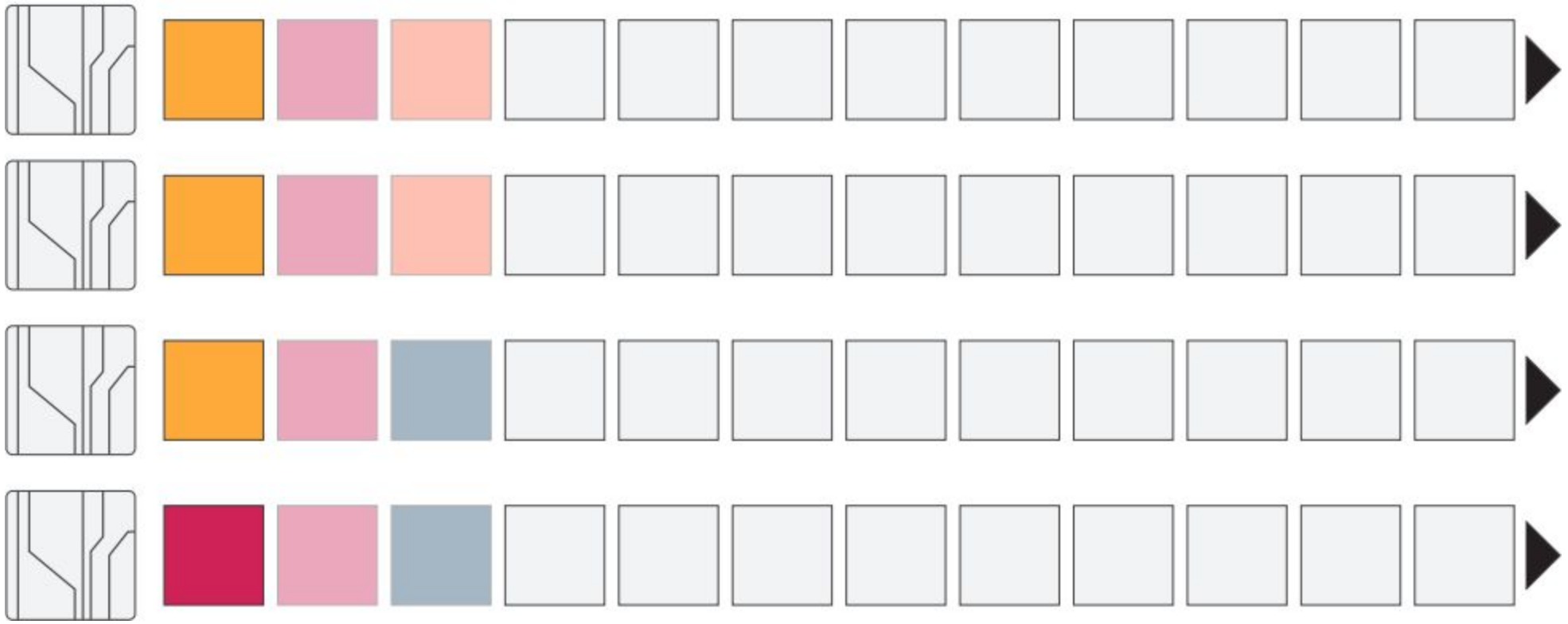
Ordonnancement parallèle



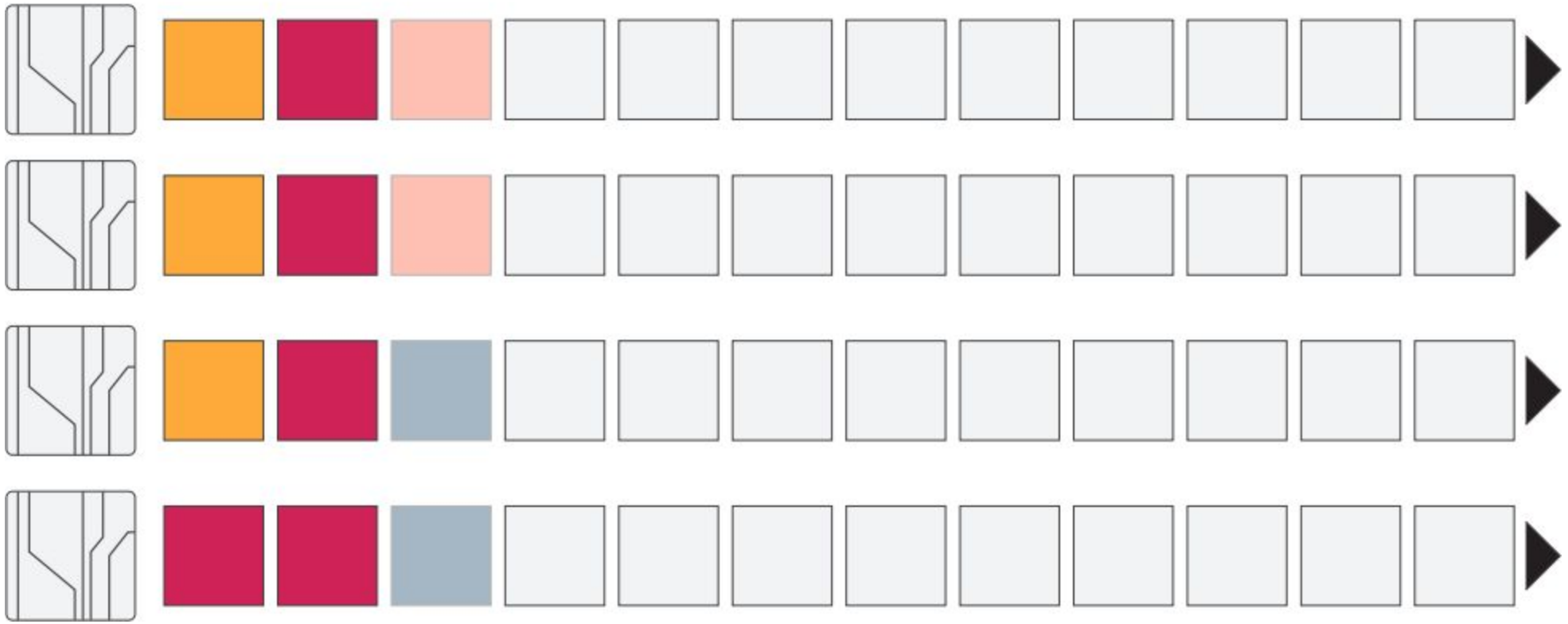
Ordonnancement parallèle



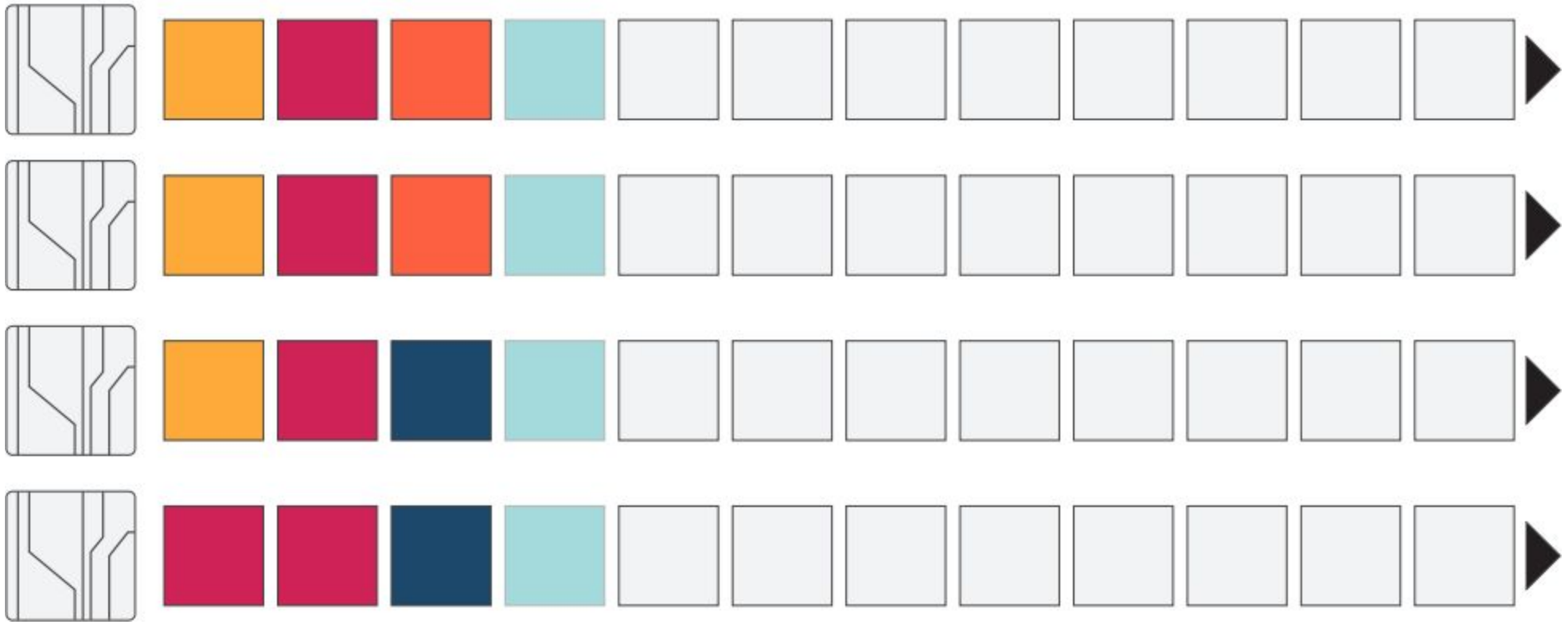
Ordonnancement parallèle



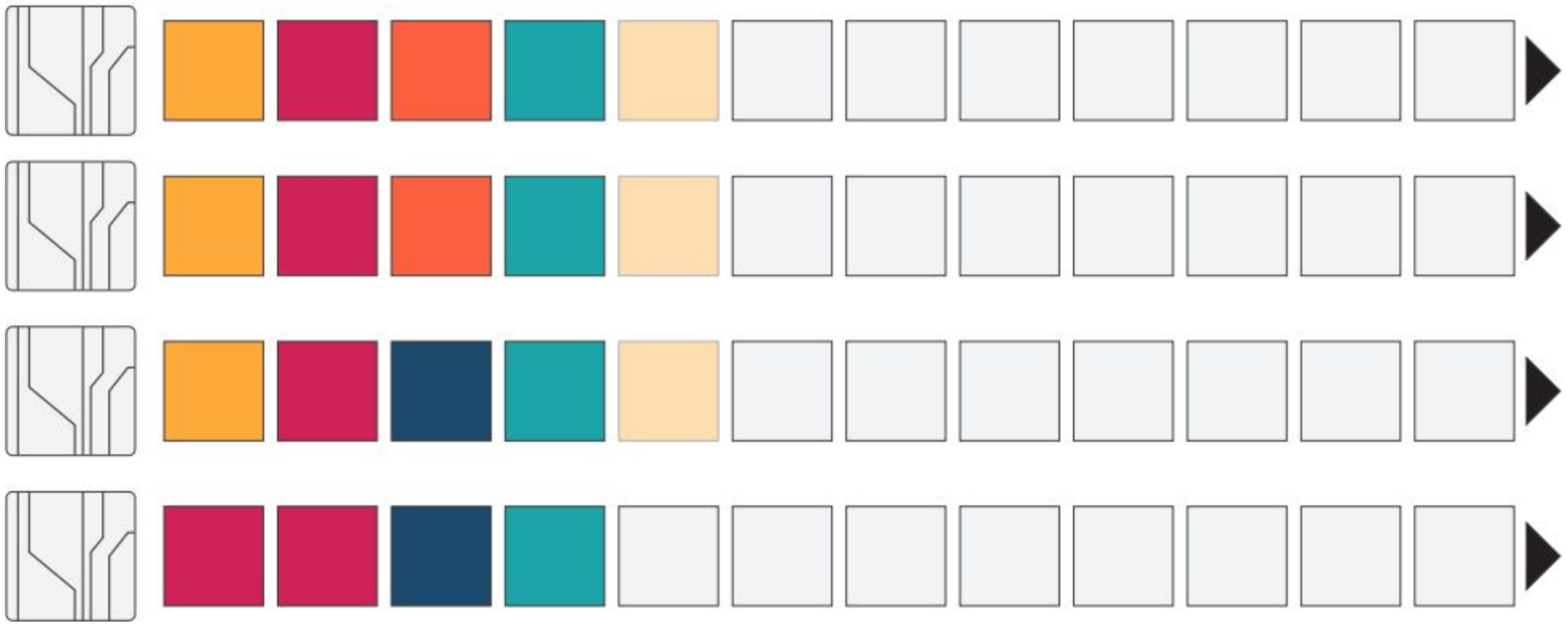
Ordonnancement parallèle



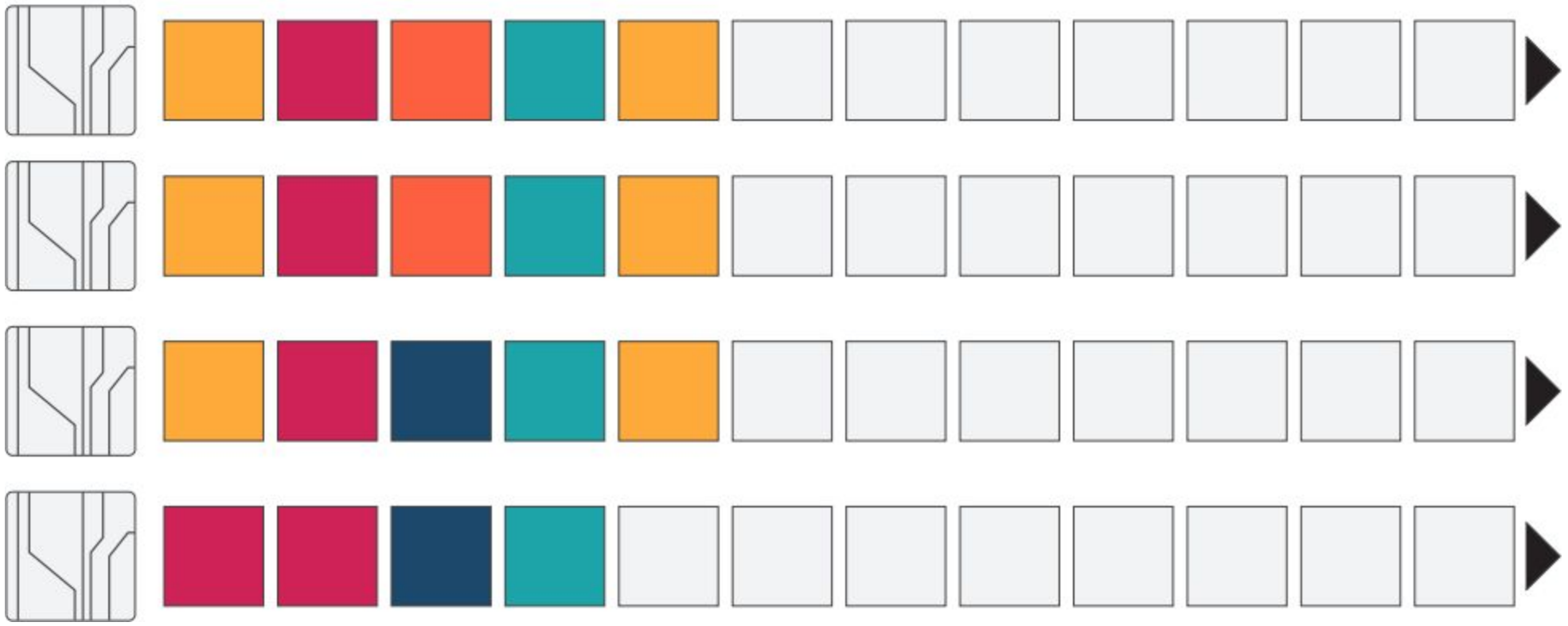
Ordonnancement parallèle



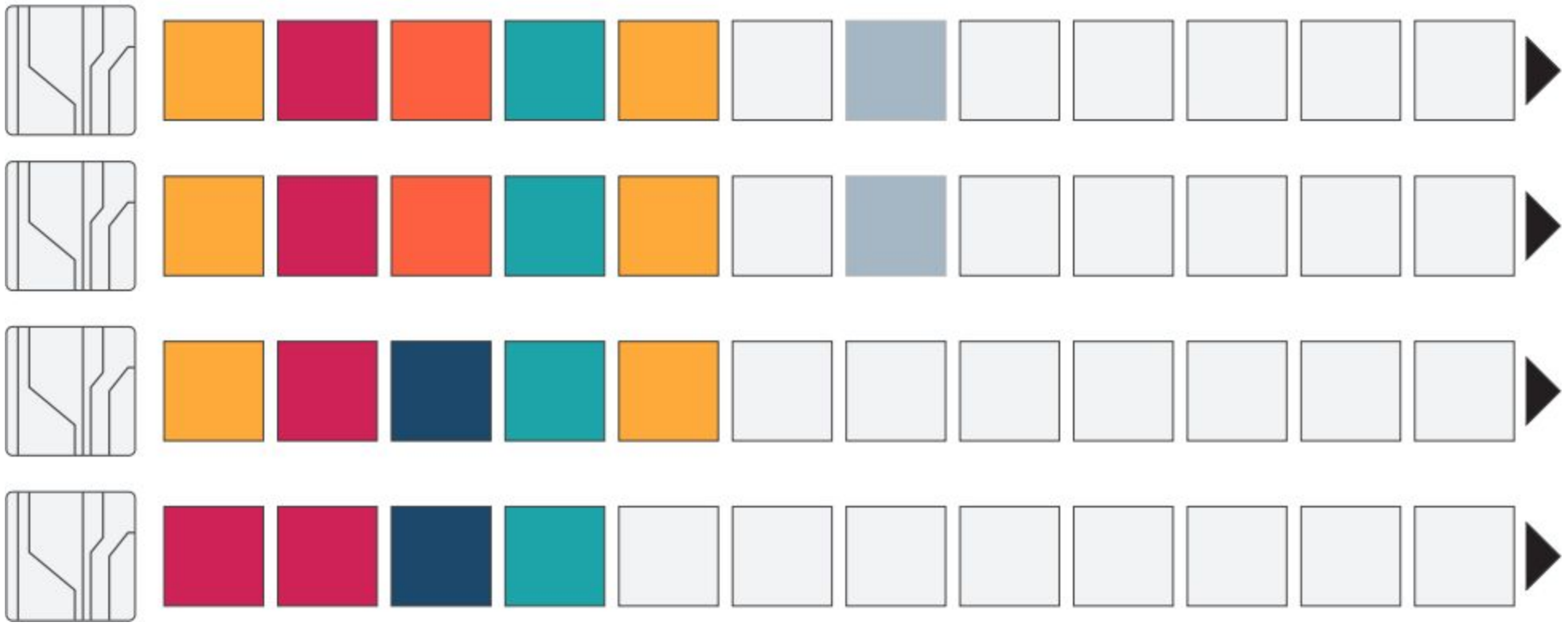
Ordonnancement parallèle



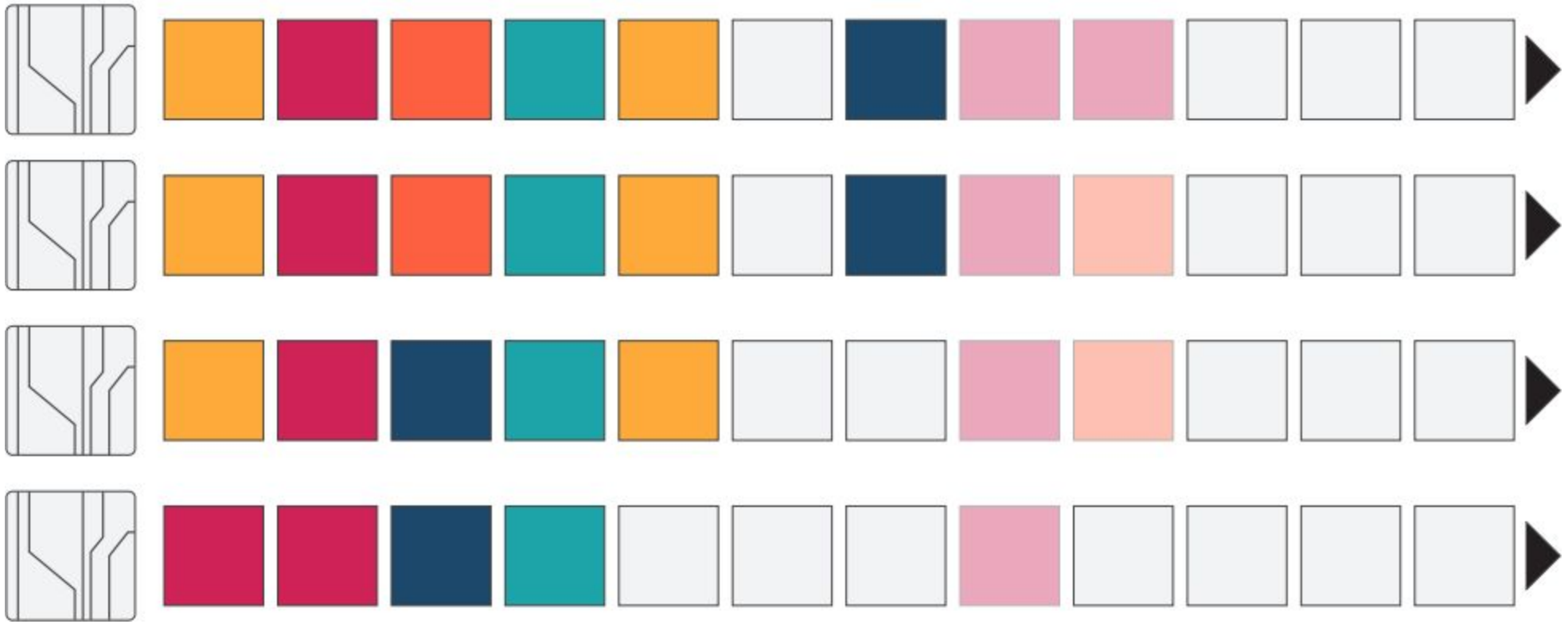
Ordonnancement parallèle



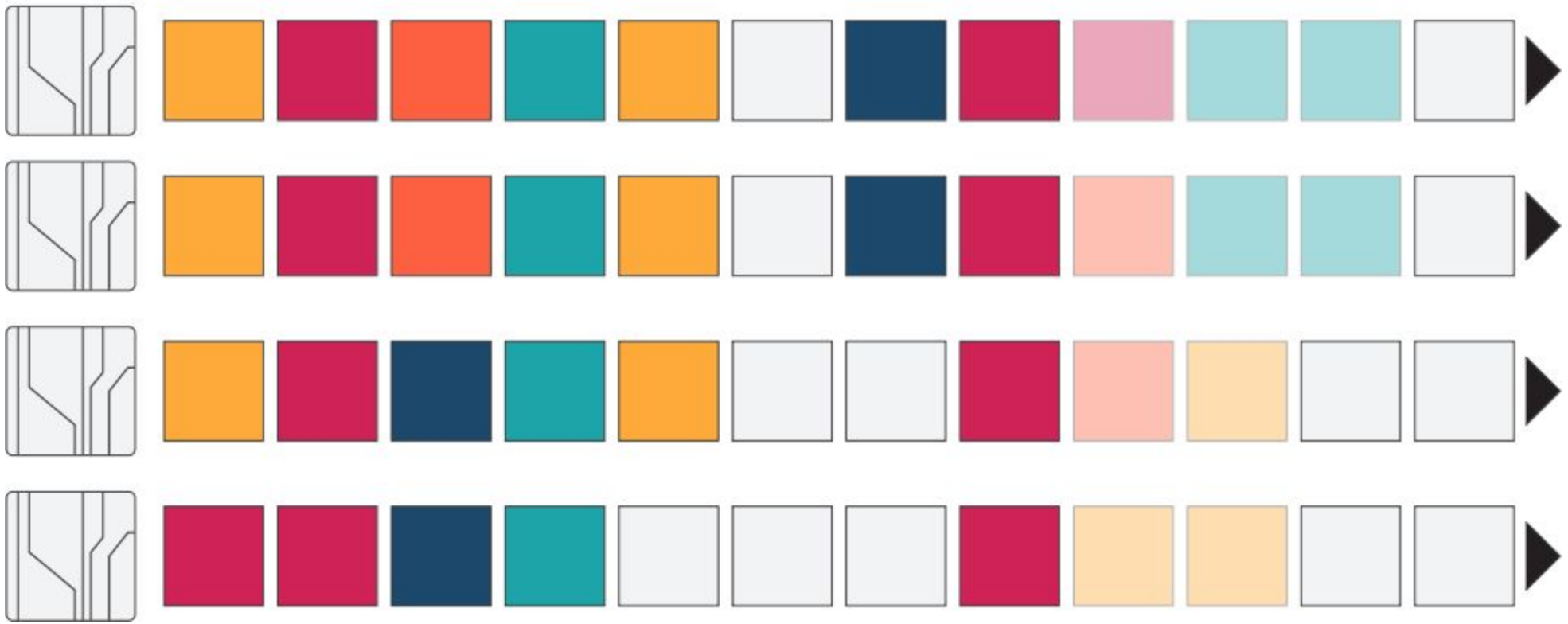
Ordonnancement parallèle



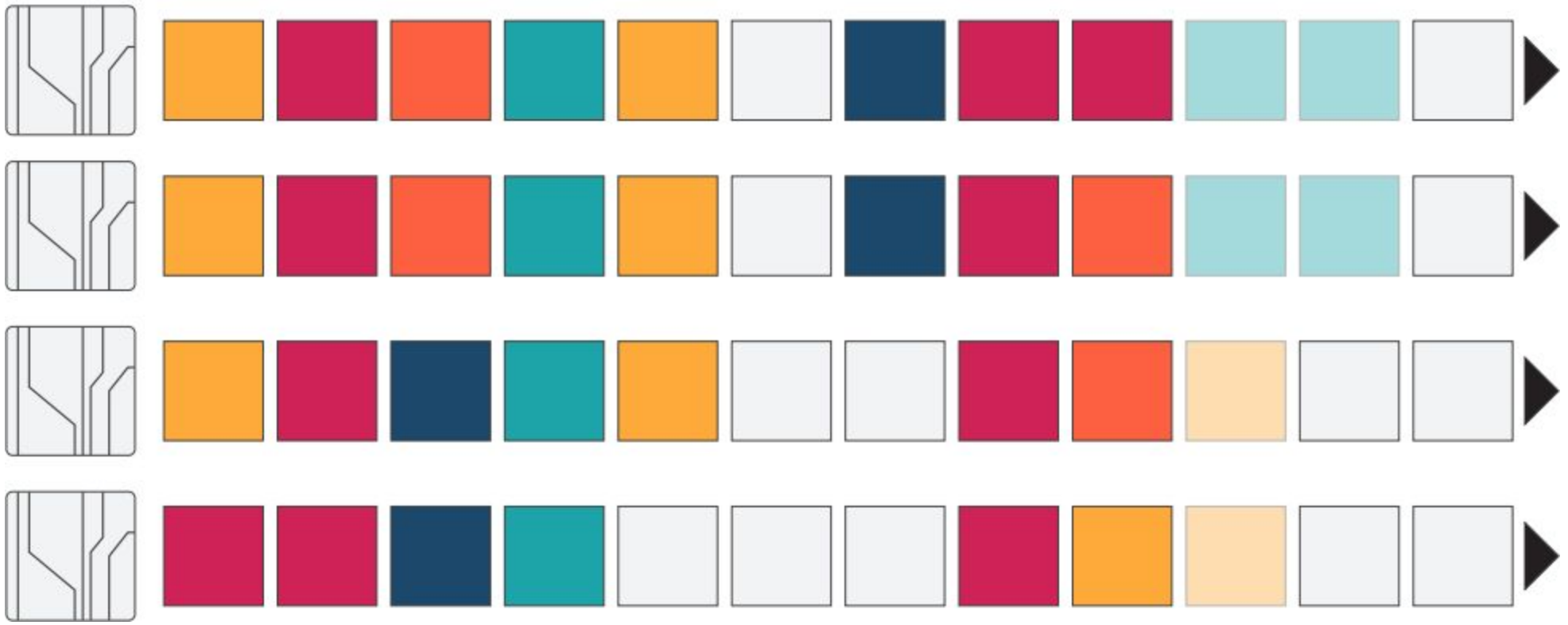
Ordonnancement parallèle



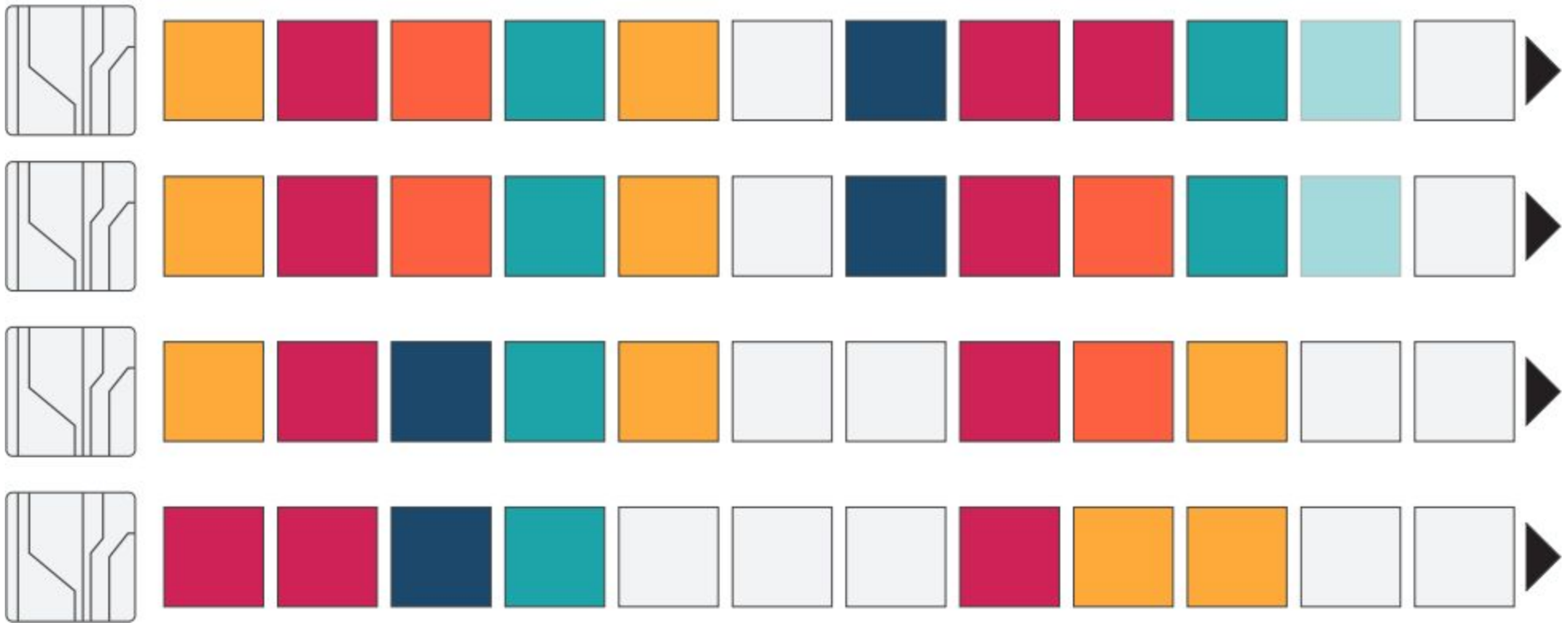
Ordonnancement parallèle



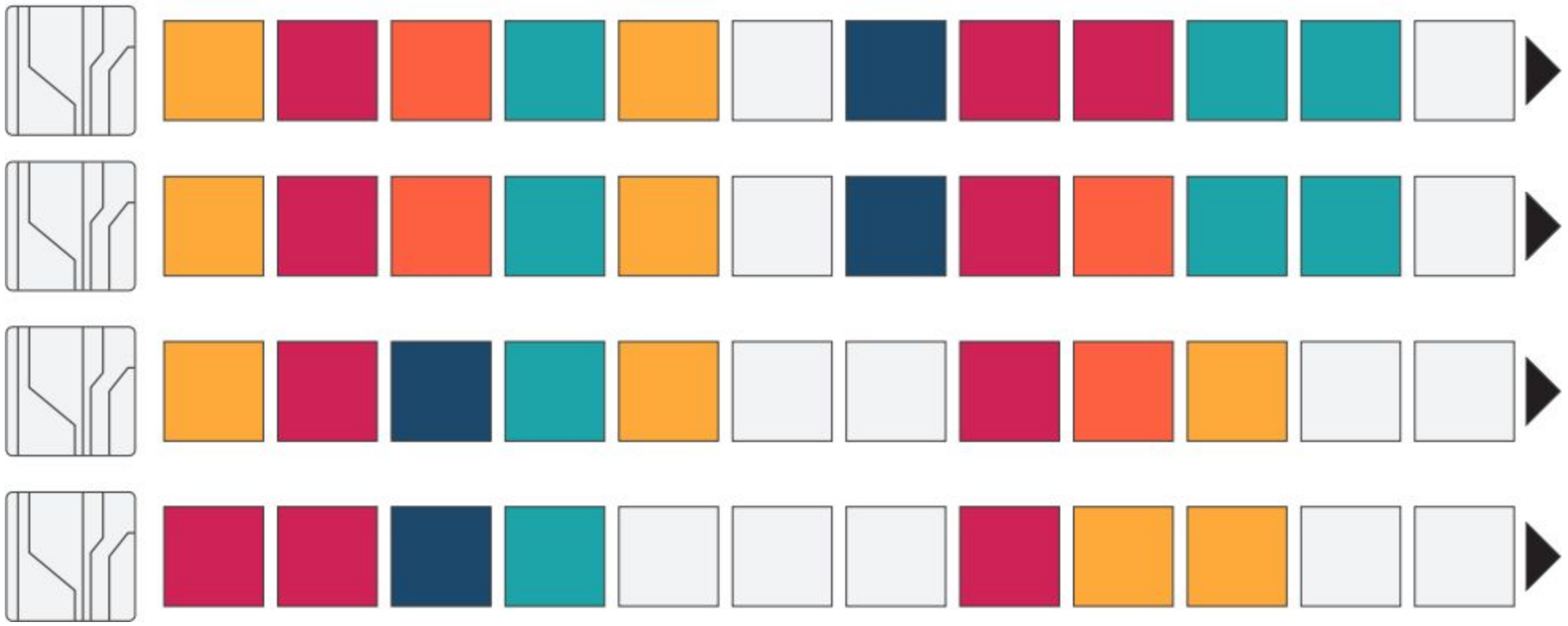
Ordonnancement parallèle



Ordonnancement parallèle



Ordonnancement parallèle



Peut-on **mieux réduire**
la consommation
en exploitant le **parallélisme** ?

Déclaration de thèse≃ solution du problème:

Déclaration de thèse≃ solution du problème:

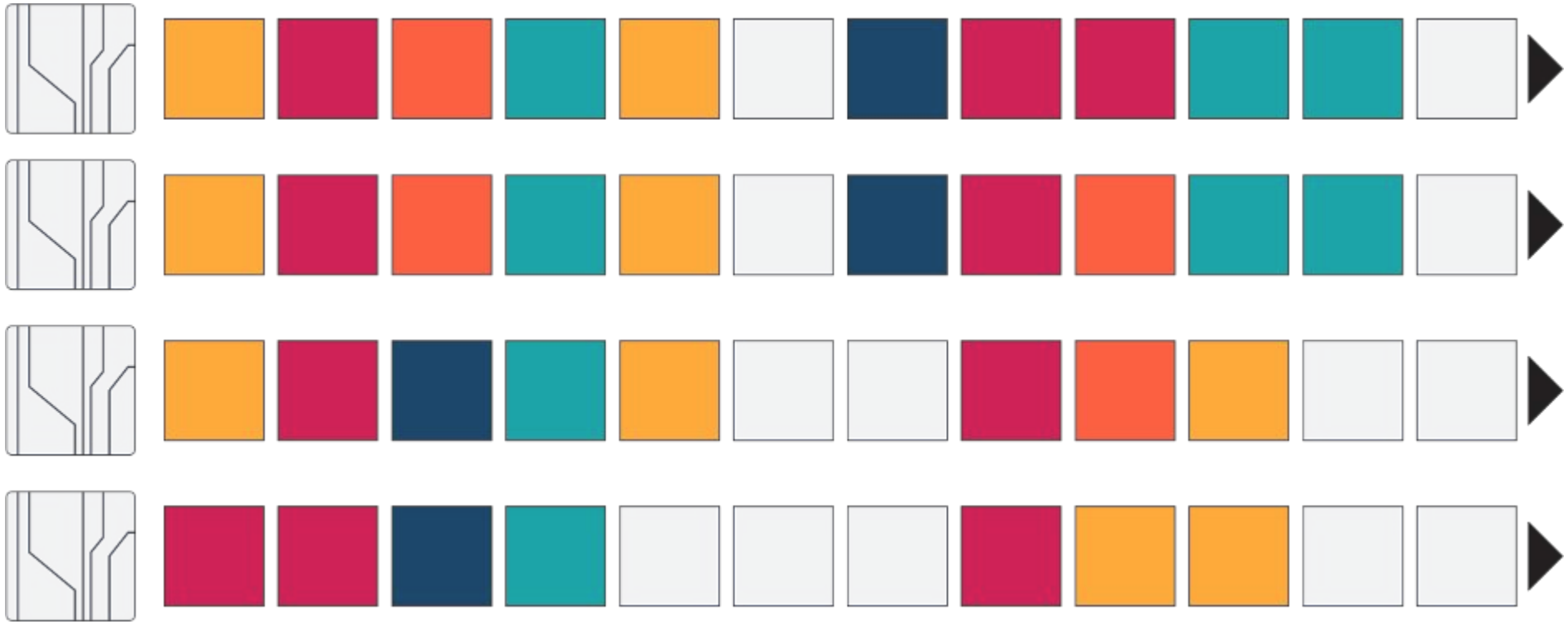
Le parallélisme permet de
réduire l'énergie consommée tout en
garantissant **les contraintes temporelles**

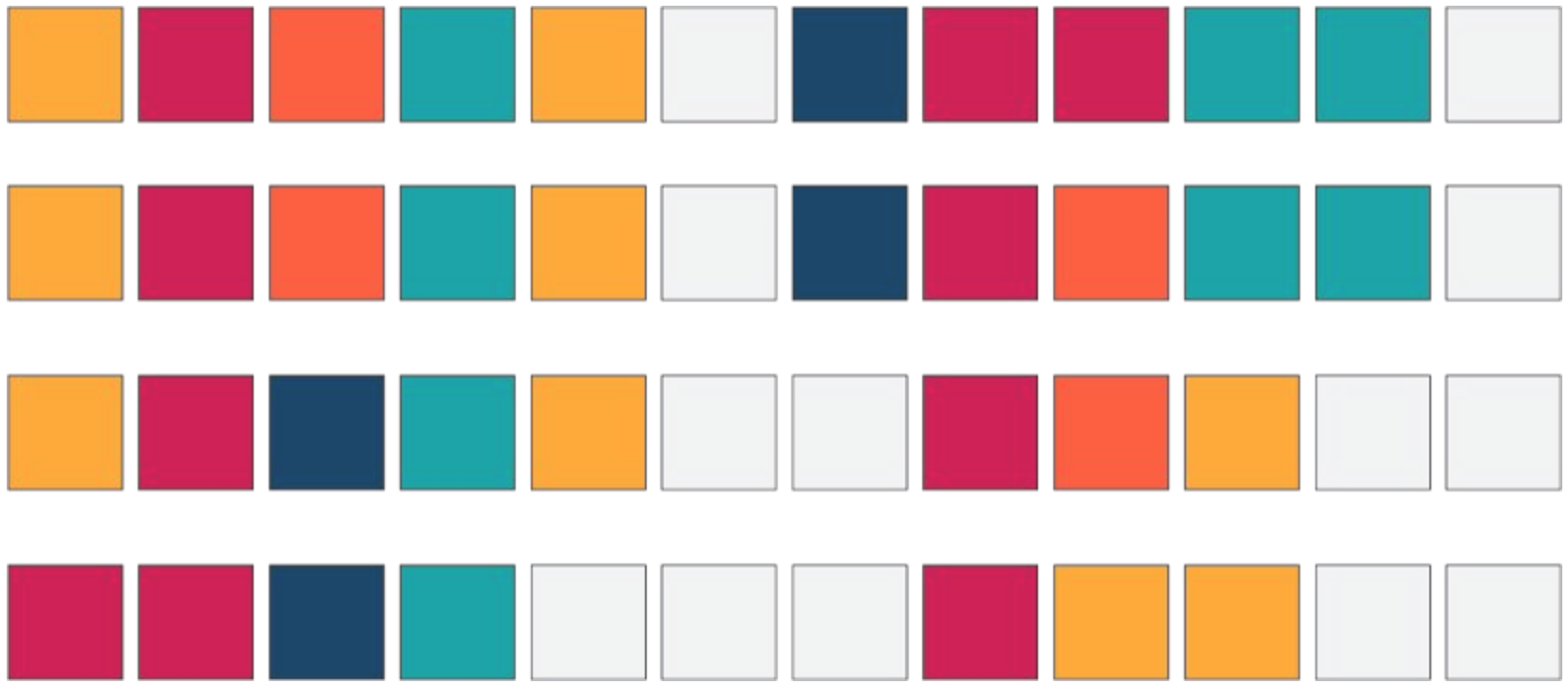
Contributions choisies

- Validation théorique
- Un nouveau système d’exploitation
- Validation pratique

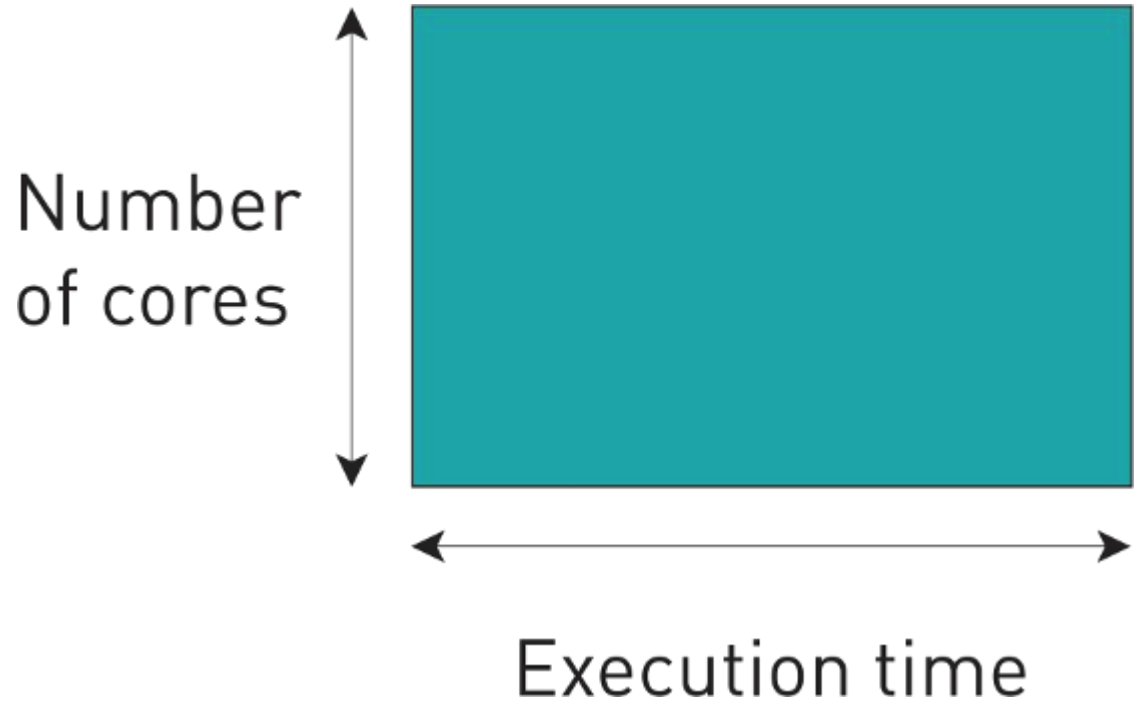
Contributions choisies

- **Validation théorique**
- Un nouveau système d’exploitation
- Validation pratique

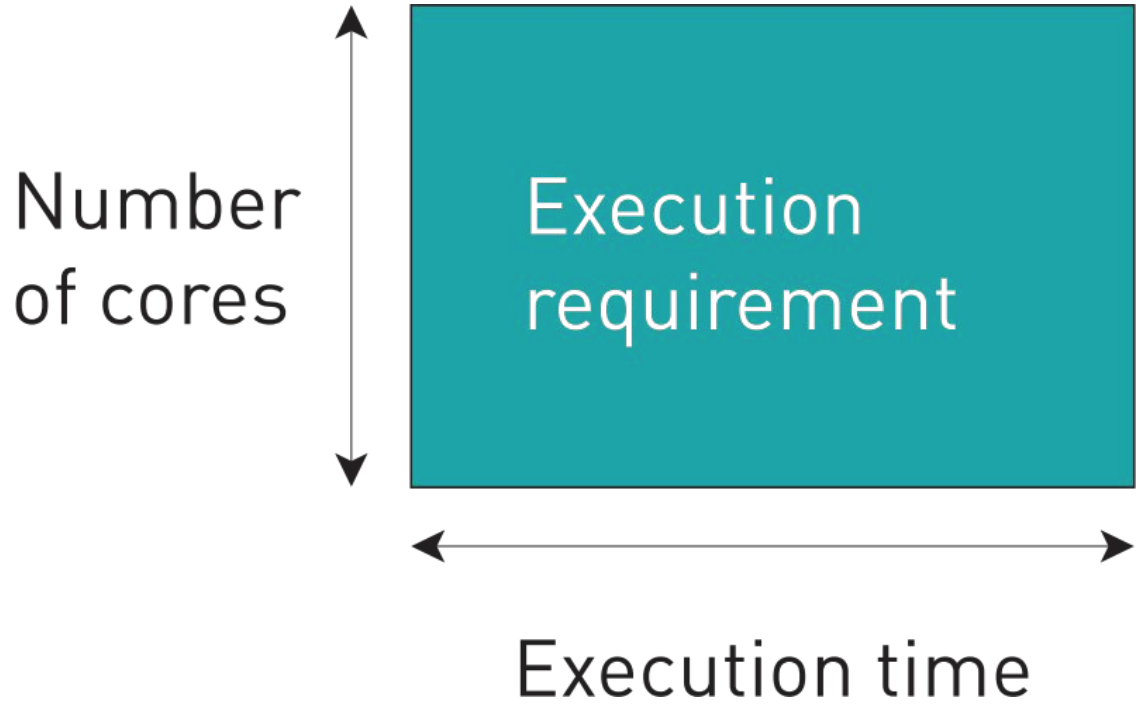




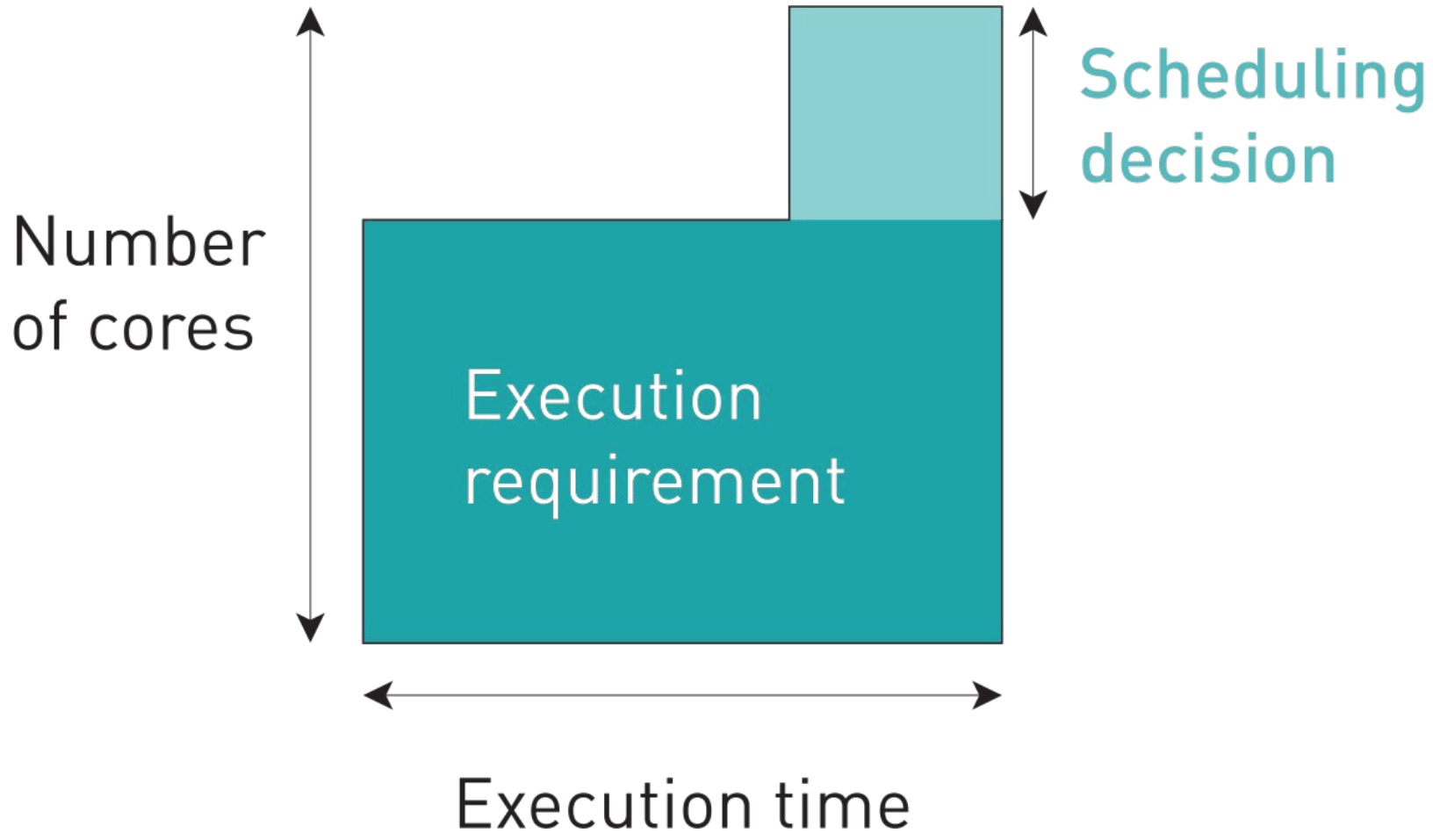
Parallel job model



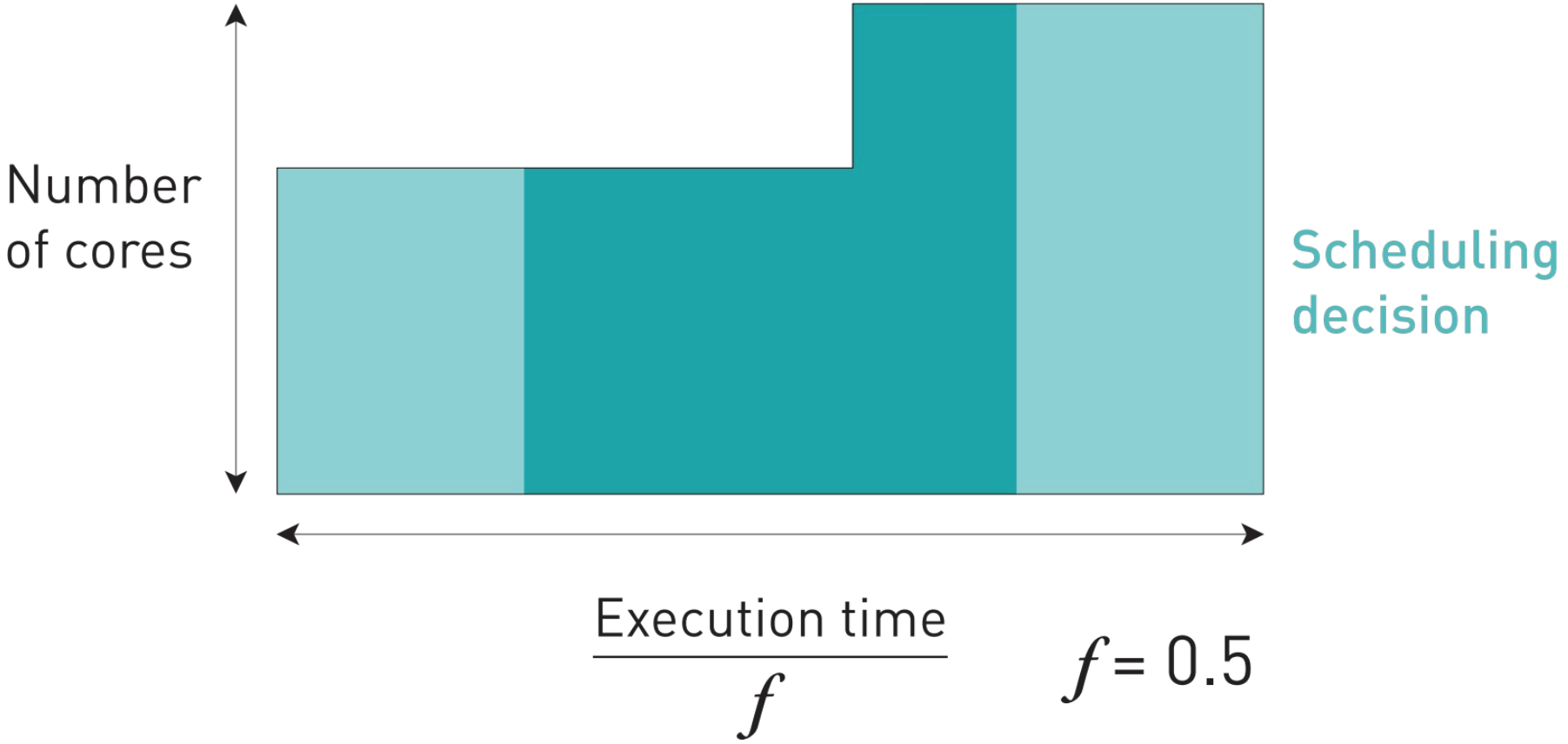
Parallel job model



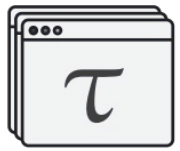
Malleable job



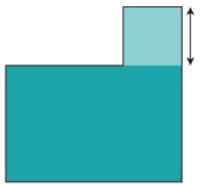
Malleable job



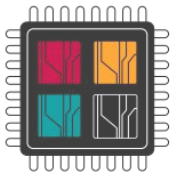
Find f and k such that $P(f, k)$ is minimized



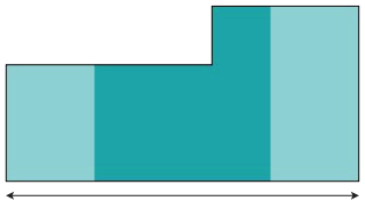
Implicit-deadline sporadic tasks



Malleable jobs



DVFS/DPM-enabled processor with m cores



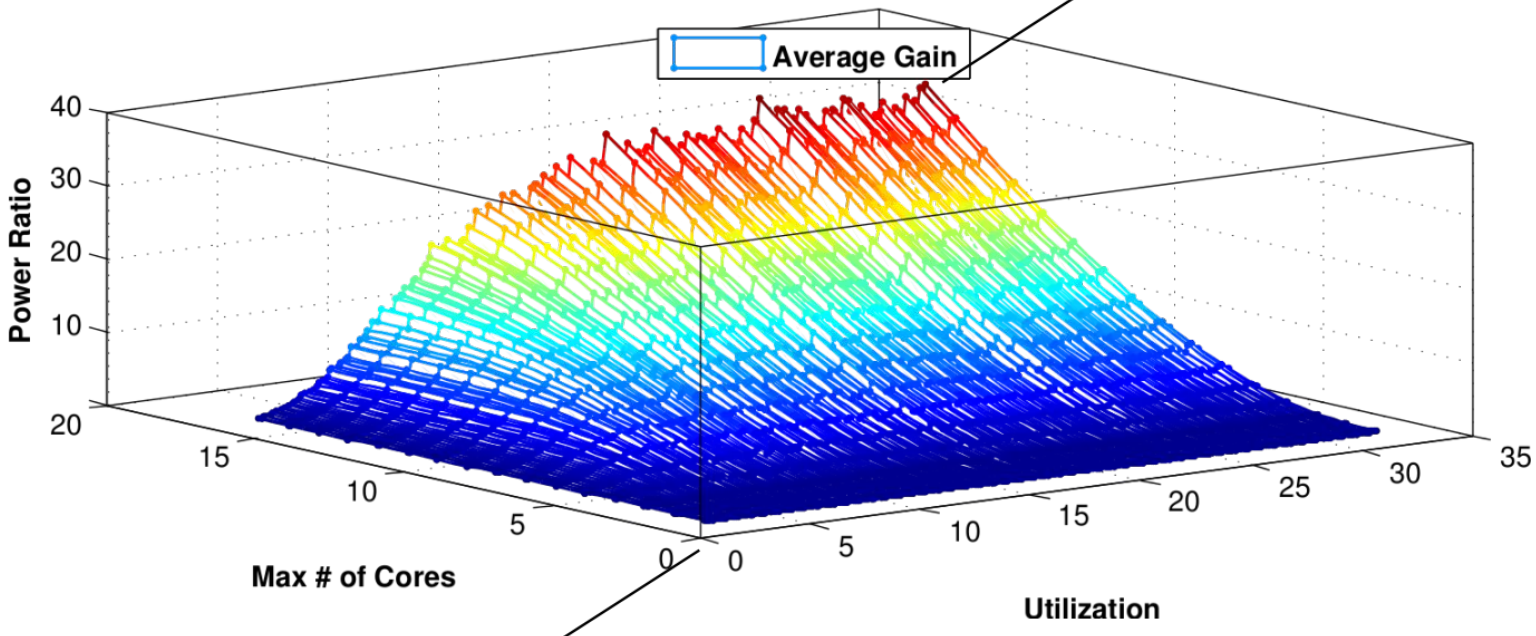
Homogeneous frequency

Canonical optimal scheduling

Results - parallelism helps

$P_{mal} << P_{seq}$

Ratio ≃ 36



$P_{mal} = P_{seq}$

Ratio = 1

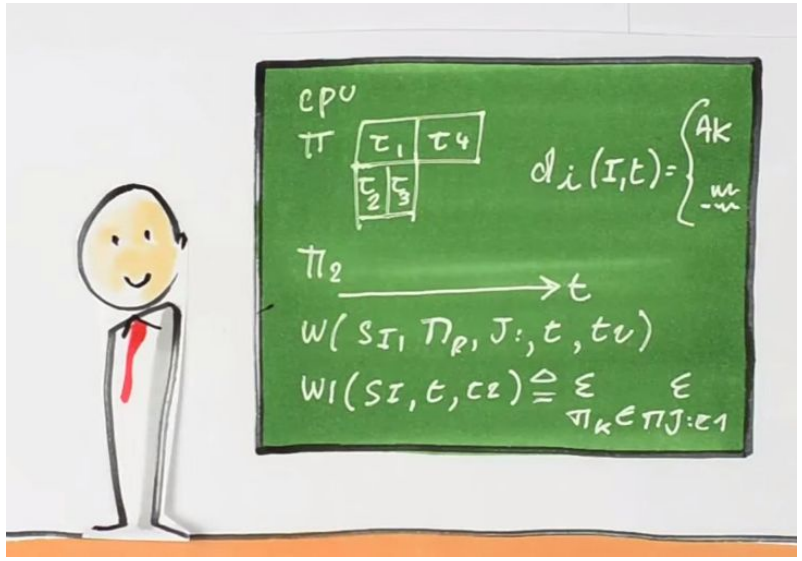
Contributions choisies

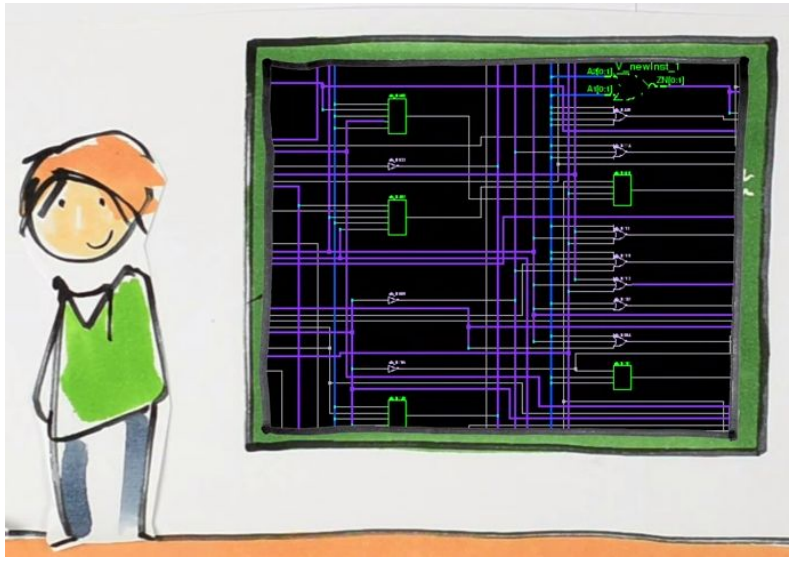
- Validation théorique
- **Un nouveau système d’exploitation**
- Validation pratique

OS pour l’embarqué en 2006

- Peu de considérations pour le multi-cœur
- Noyaux actuels ne passent pas à l’échelle [Cerqueira2014]
- Contraintes temporelles et gestion de l’alimentation non intégrés
- **Ordonnancement:** écart entre la recherche et industrie

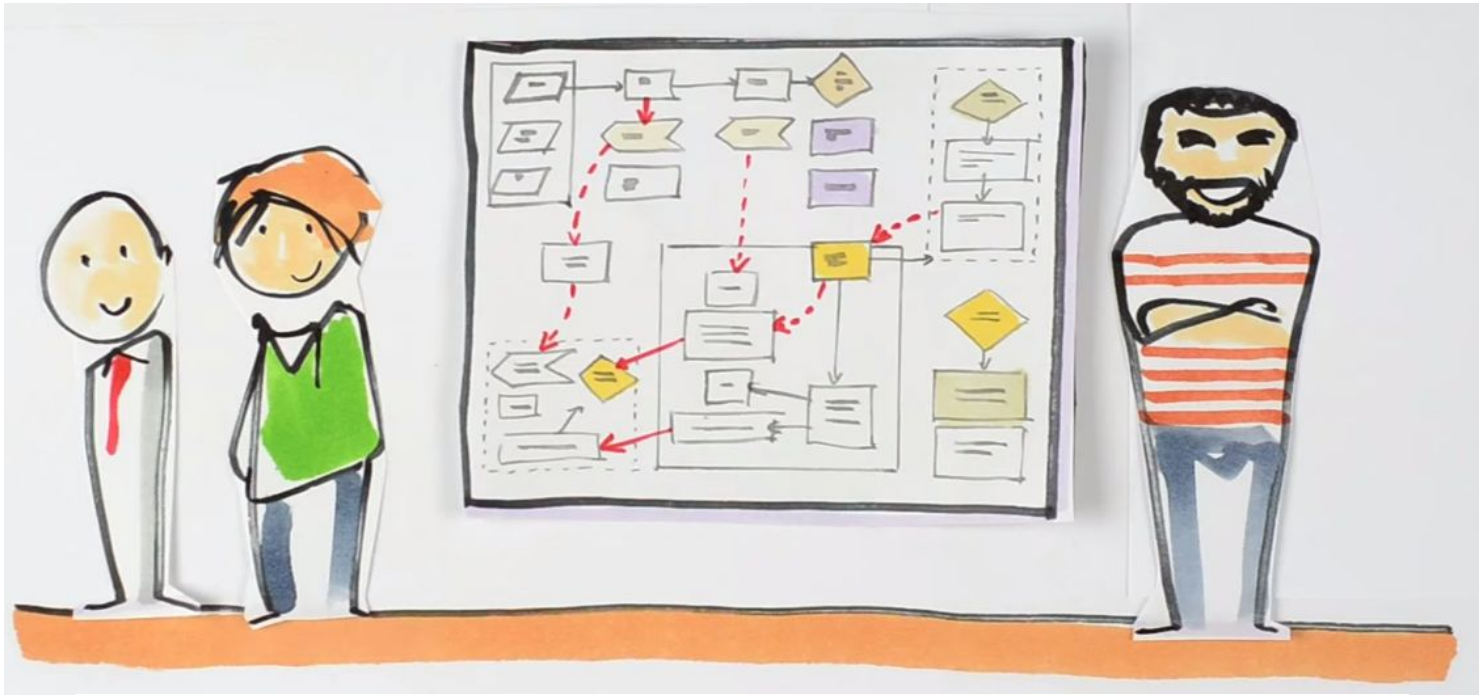
Parallel Architecture for Real-Time Systems





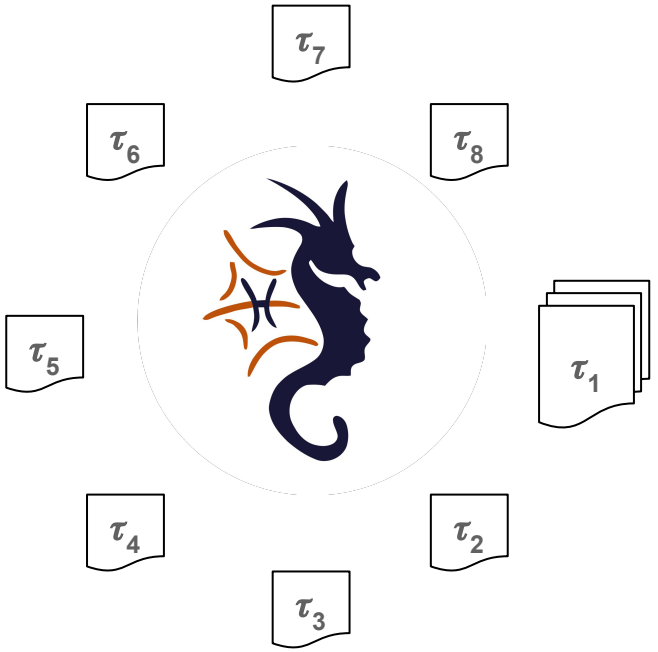
PARTS

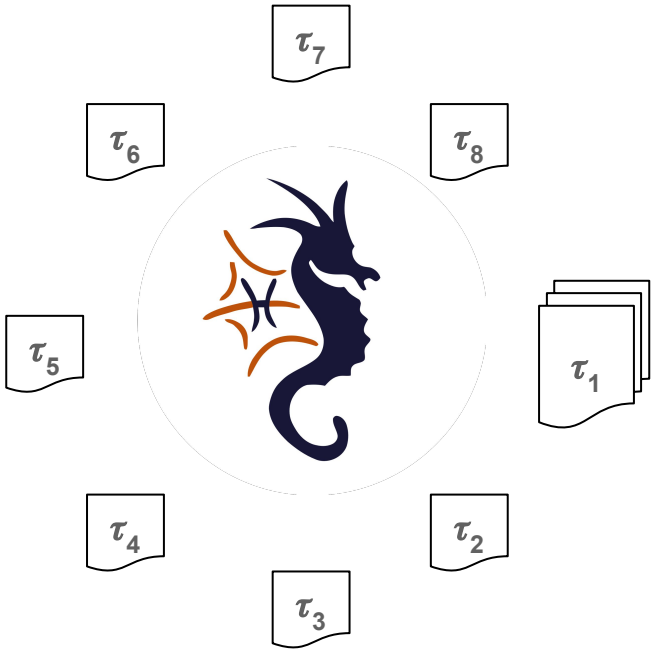
Parallel Architecture for Real-Time Systems

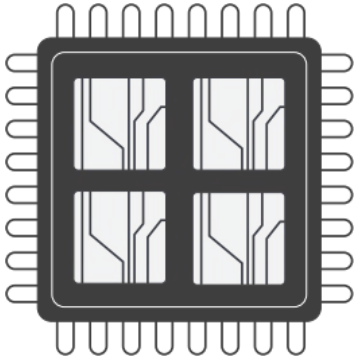


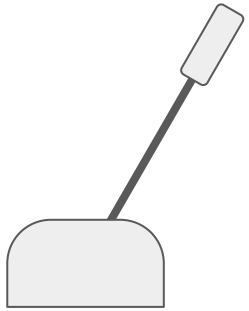
HIPPEROS®

Predictable Real-Time, Proven Performance


τ_7
τ_6
τ_8
τ_5
τ_1
τ_4
τ_2
τ_3


τ_7
τ_6
τ_8
τ_5
τ_1
τ_4
τ_2
τ_3





Contributions choisies

- Validation théorique
- Un nouveau système d’exploitation
- **Validation pratique**

Plate-forme expérimentale

Carte embarquée

Plate-forme expérimentale

Carte embarquée

Plate-forme expérimentale

OS
& bibliothèques

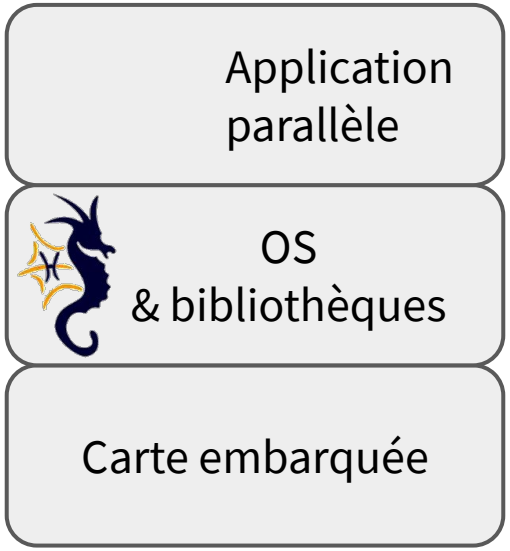
Carte embarquée

Plate-forme expérimentale

OS
& bibliothèques

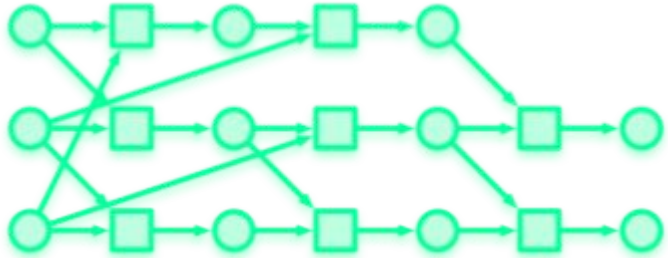
Carte embarquée

Plate-forme expérimentale



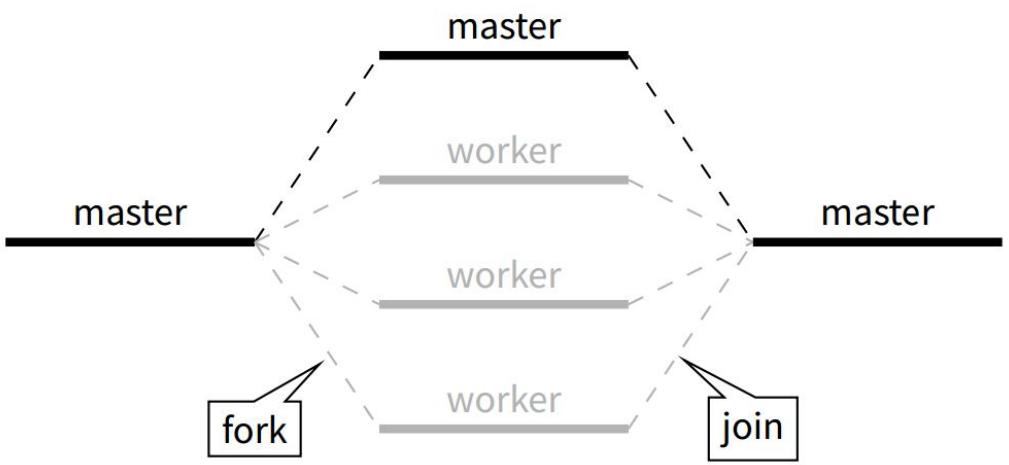
Application parallèle ?

OPTIMISATION
OF PERFORMANCE METRICS
OF EMBEDDED
HARD REAL-TIME SYSTEMS
USING **SOFTWARE**/HARDWARE
PARALLELISM

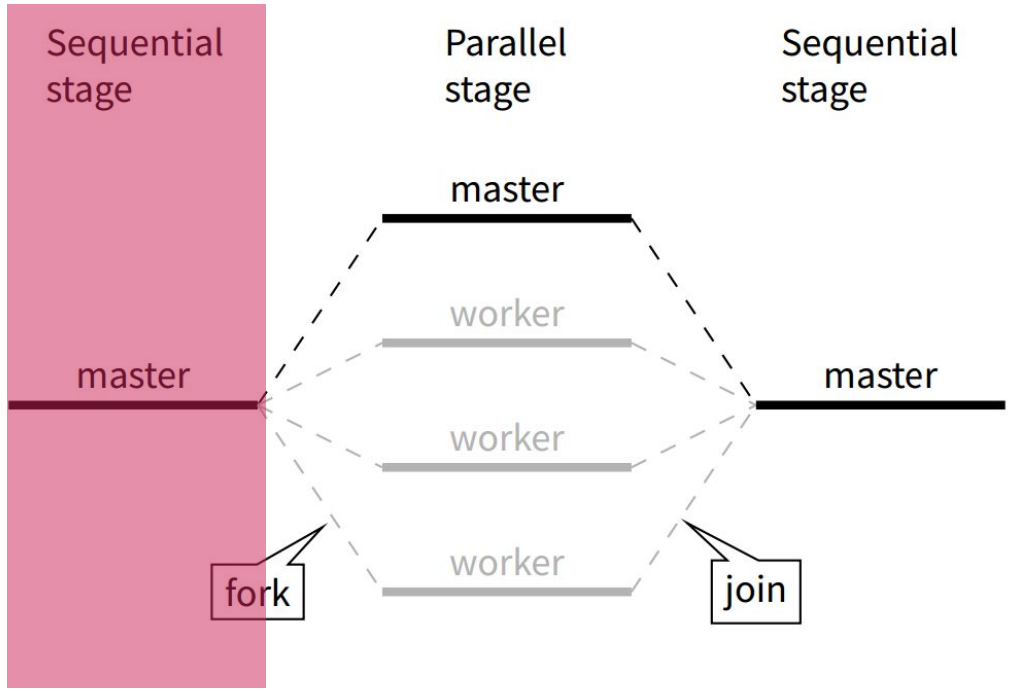


Application parallèle ?

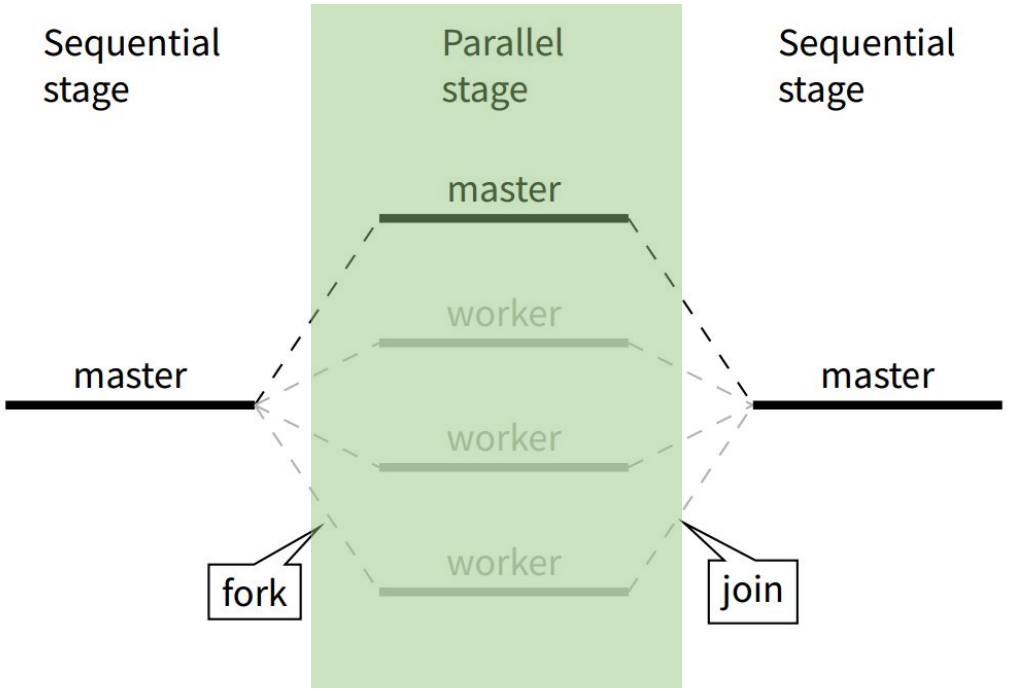
```
1  int main()
2  {
3    const int num_steps = 1000;
4    omp_set_num_threads(4);
5
6  #pragma omp parallel
7    {
8      int nbt = omp_get_num_threads();
9      int tid = omp_get_thread_num();
10     int i_start = (tid * num_steps) / nbt;
11     int i_end = ((tid + 1) * num_steps) / nbt;
12
13     for (int i = i_start; i < i_end; ++i) {
14       /* workload executed in parallel */
15     }
16   }
17
18   return 0;
19 }
```



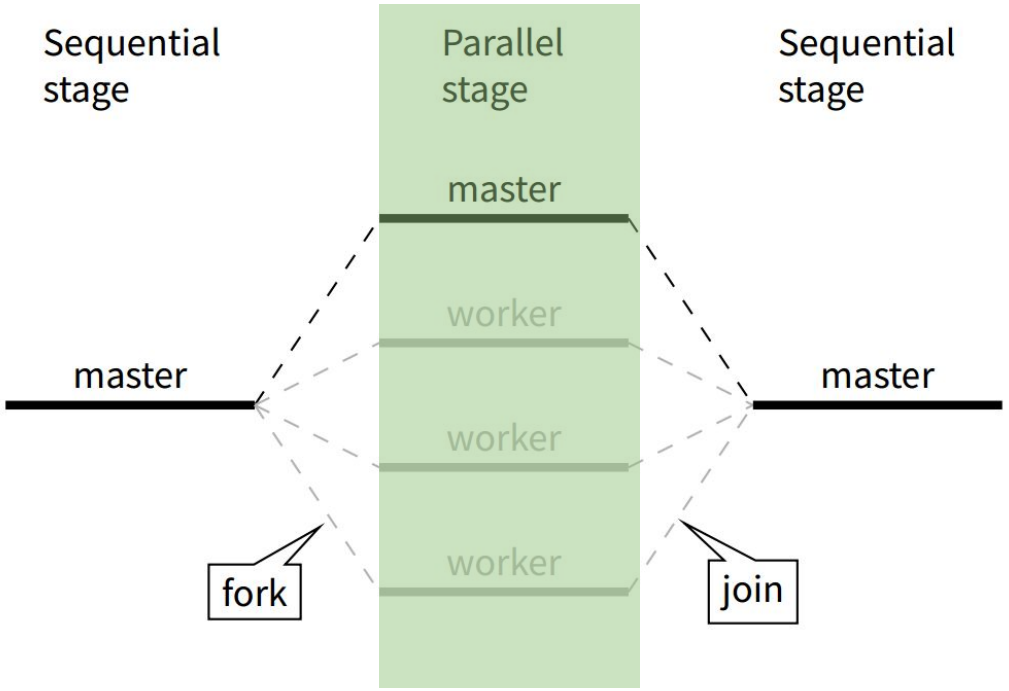

```
1  int main()
2  {
3    const int num_steps = 1000;
4    omp_set_num_threads(4);
5
6  #pragma omp parallel
7    {
8      int nbt = omp_get_num_threads();
9      int tid = omp_get_thread_num();
10     int i_start = (tid * num_steps) / nbt;
11     int i_end = ((tid + 1) * num_steps) / nbt;
12
13     for (int i = i_start; i < i_end; ++i) {
14       /* workload executed in parallel */
15     }
16   }
17
18   return 0;
19 }
```



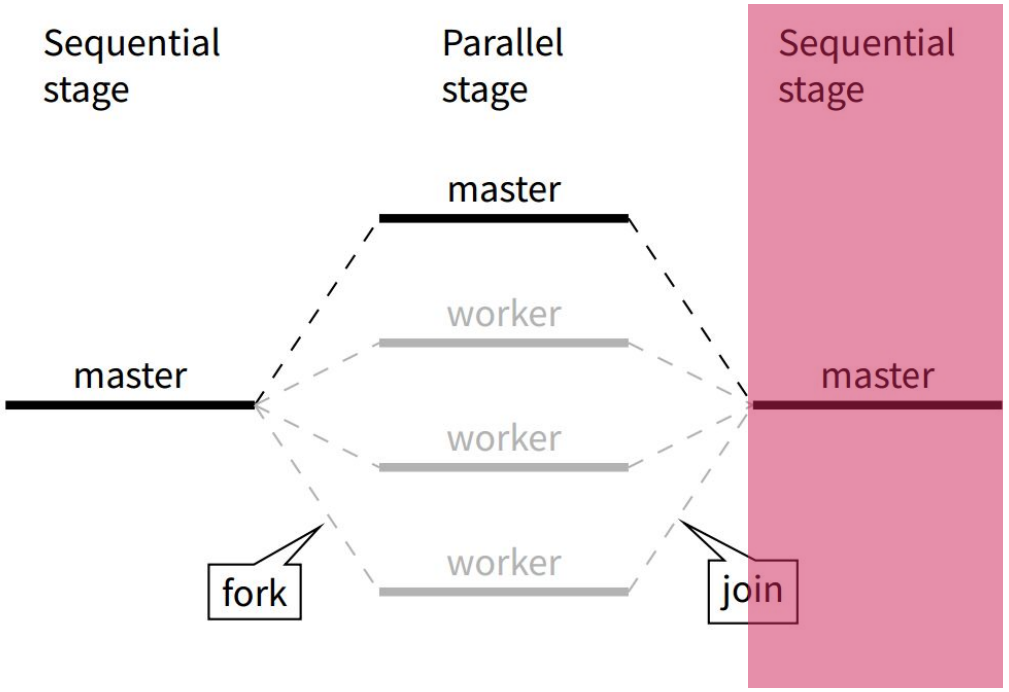

```c
int main()
{
  const int num_steps = 1000;
  omp_set_num_threads(4);

#pragma omp parallel
  {
    int nbt = omp_get_num_threads();
    int tid = omp_get_thread_num();
    int i_start = (tid * num_steps) / nbt;
    int i_end = ((tid + 1) * num_steps) / nbt;

    for (int i = i_start; i < i_end; ++i) {
      /* workload executed in parallel */
    }
  }

  return 0;
}
```



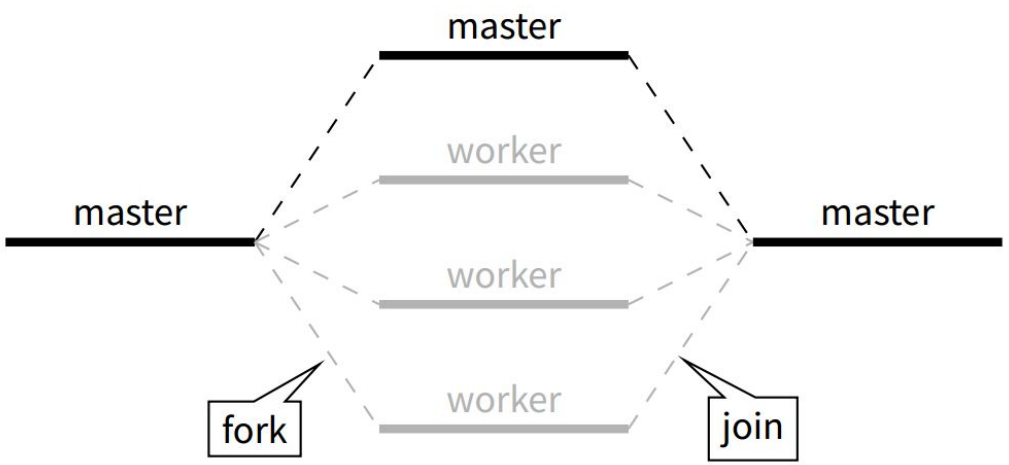

```c
int main()
{
  const int num_steps = 1000;
  omp_set_num_threads(4);

#pragma omp parallel
  {
    int nbt = omp_get_num_threads();
    int tid = omp_get_thread_num();
    int i_start = (tid * num_steps) / nbt;
    int i_end = ((tid + 1) * num_steps) / nbt;

    for (int i = i_start; i < i_end; ++i) {
      /* workload executed in parallel */
    }
  }

  return 0;
}
```




```
1  int main()
2  {
3    const int num_steps = 1000;
4    omp_set_num_threads(4);
5
6  #pragma omp parallel
7    {
8      int nbt = omp_get_num_threads();
9      int tid = omp_get_thread_num();
10     int i_start = (tid * num_steps) / nbt;
11     int i_end = ((tid + 1) * num_steps) / nbt;
12
13     for (int i = i_start; i < i_end; ++i) {
14       /* workload executed in parallel */
15     }
16   }
17
18   return 0;
19 }
```




```
1  int main()
2  {
3    const int num_steps = 1000;
4    omp_set_num_threads(4);
5
6  #pragma omp parallel
7    {
8      int nbt = omp_get_num_threads();
9      int tid = omp_get_thread_num();
10     int i_start = (tid * num_steps) / nbt;
11     int i_end = ((tid + 1) * num_steps) / nbt;
12
13     for (int i = i_start; i < i_end; ++i) {
14       /* workload executed in parallel */
15     }
16   }
17
18   return 0;
19 }
```




```
1  int main()
2  {
3    const int num_steps = 1000;
4    omp_set_num_threads(4);
5
6  #pragma omp parallel
7    {
8      int nbt = omp_get_num_threads();
9      int tid = omp_get_thread_num();
10     int i_start = (tid * num_steps) / nbt;
11     int i_end = ((tid + 1) * num_steps) / nbt;
12
13     for (int i = i_start; i < i_end; ++i) {
14       /* workload executed in parallel */
15     }
16   }
17
18   return 0;
19 }
```

Plate-forme expérimentale

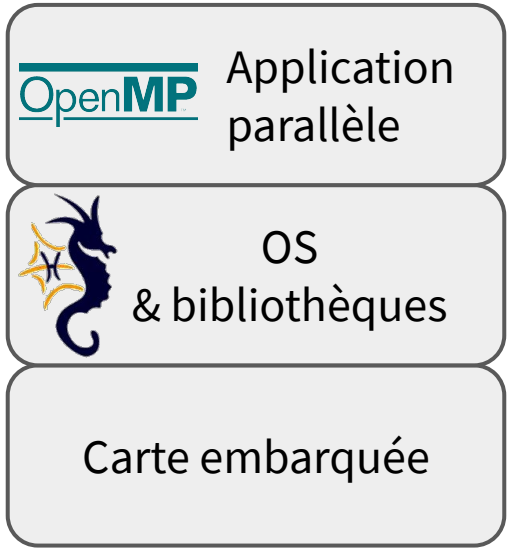
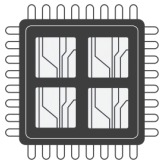
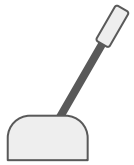
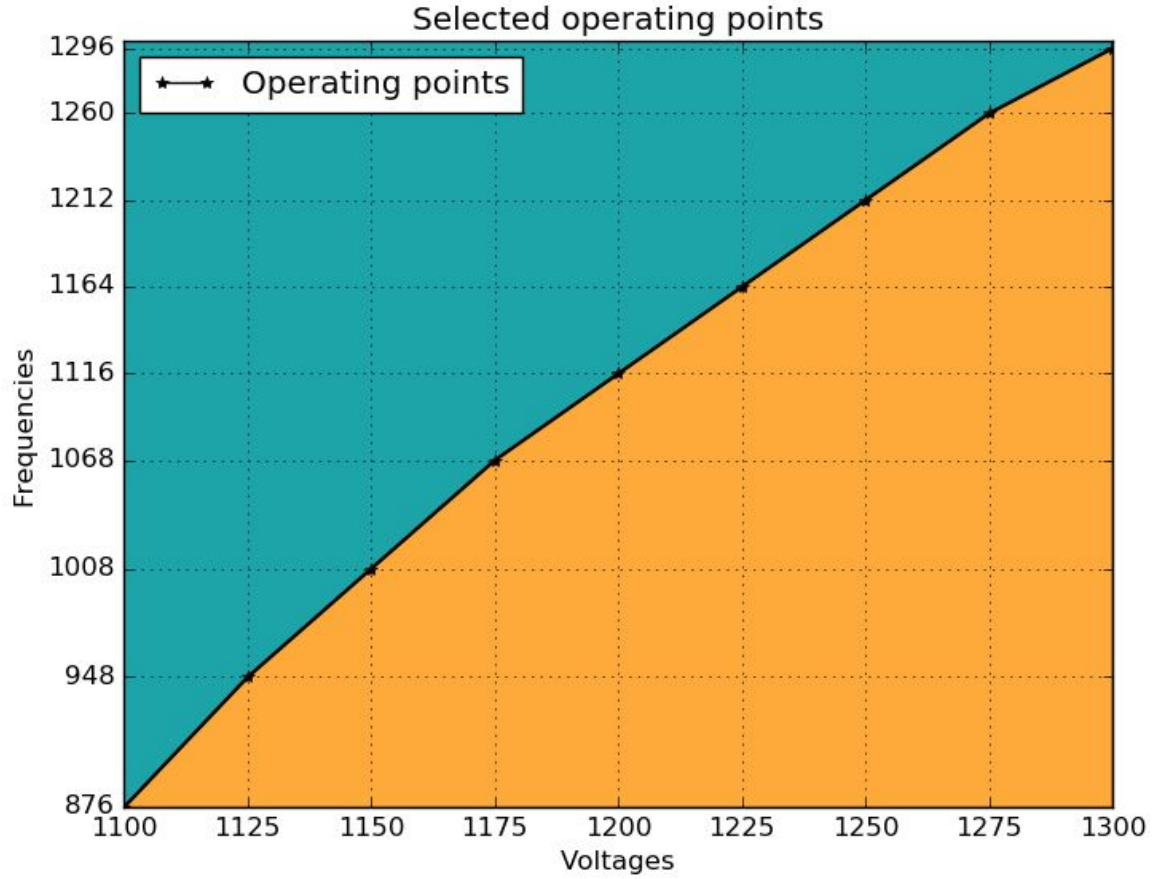


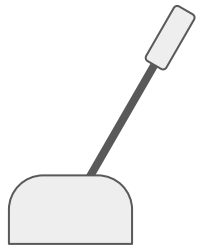
Plate-forme ciblée - i.MX6q SabreLite

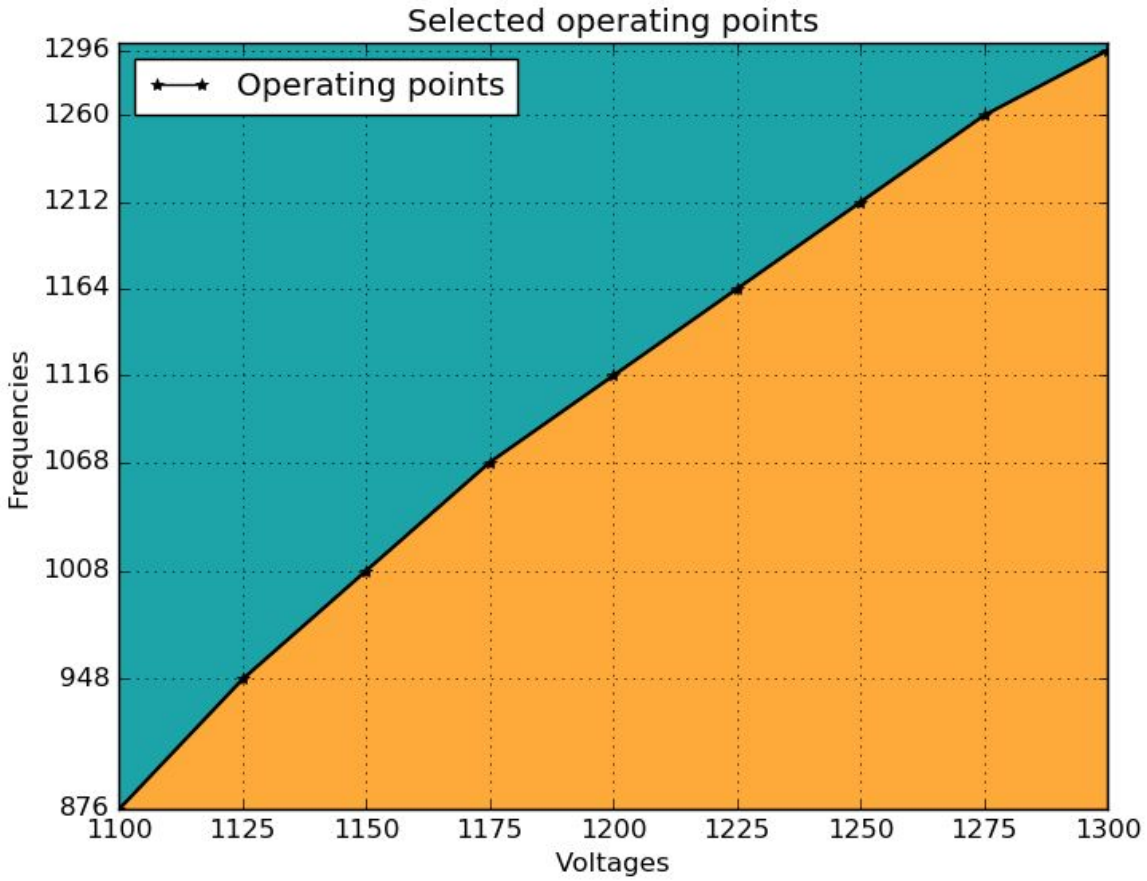
- Carte embarquée ARM Cortex A9 MP
- Supportée par HIPPEROS
- 4 cœurs 
- DVFS global *(operating points)* 

DVFS


Selected operating points
Operating points
Frequencies
1296
1260
1212
1164
1116
1068
1008
948
876
1100
1125
1150
1175
1200
1225
1250
1275
1300
Voltages

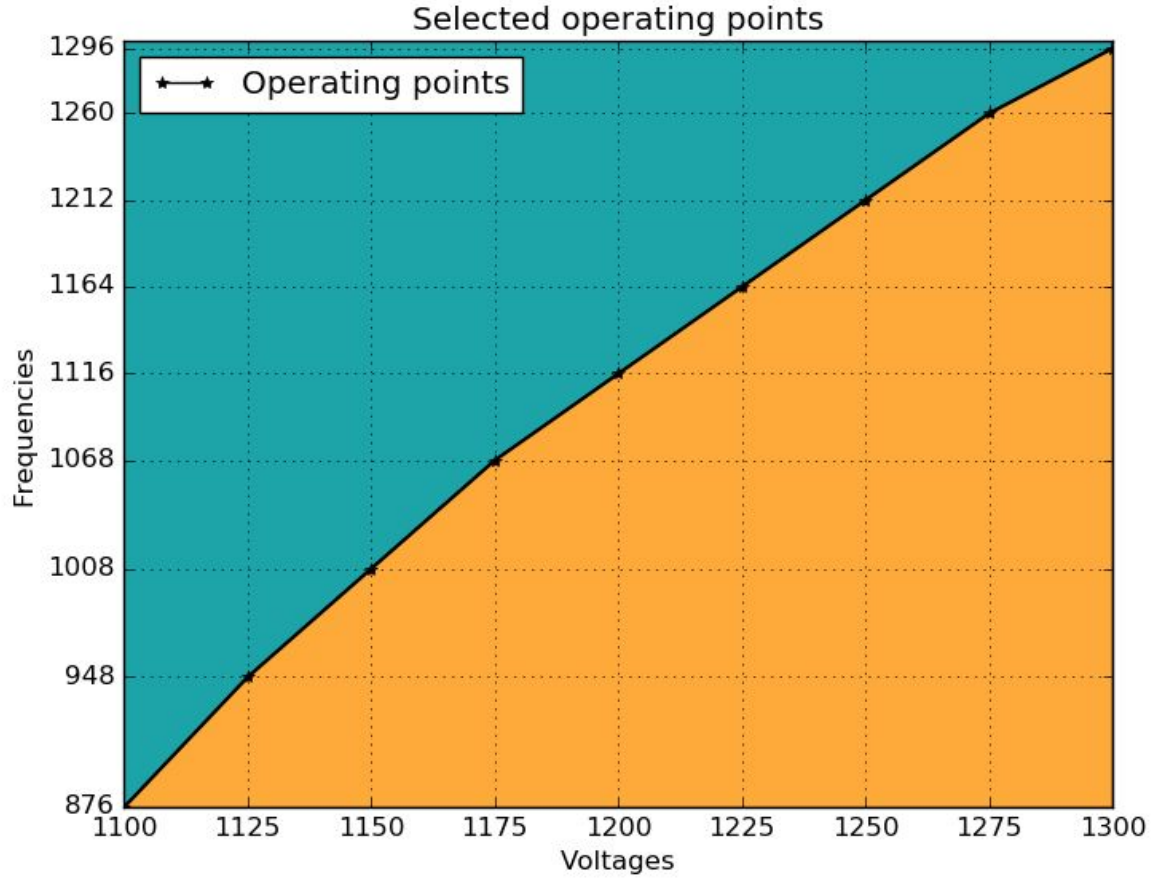
DVFS





Voltage (mV)	Frequency (MHz)
1100	876
1125	948
1150	1008
1175	1068
1200	1116
1225	1164
1250	1212
1275	1260
1300	1296

DVFS et consommation



$$P \propto V_{dd}^{2} f$$

$$V_{dd} \propto f$$

Plate-forme expérimentale

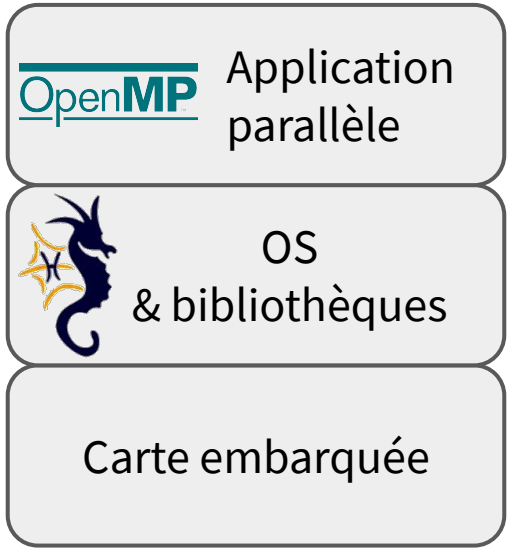


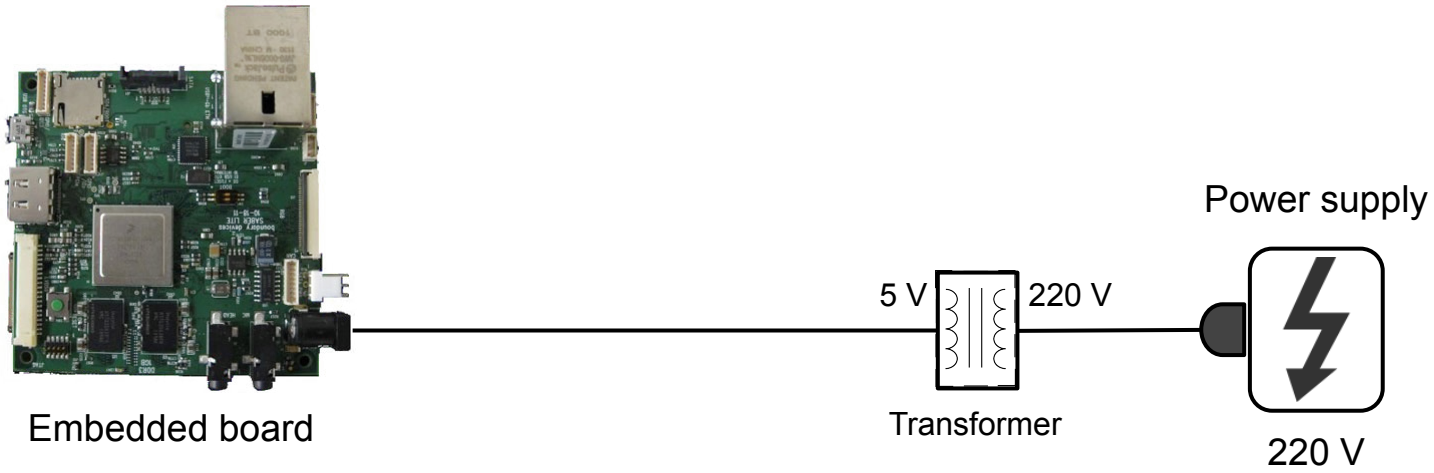
Plate-forme expérimentale

Application parallèle

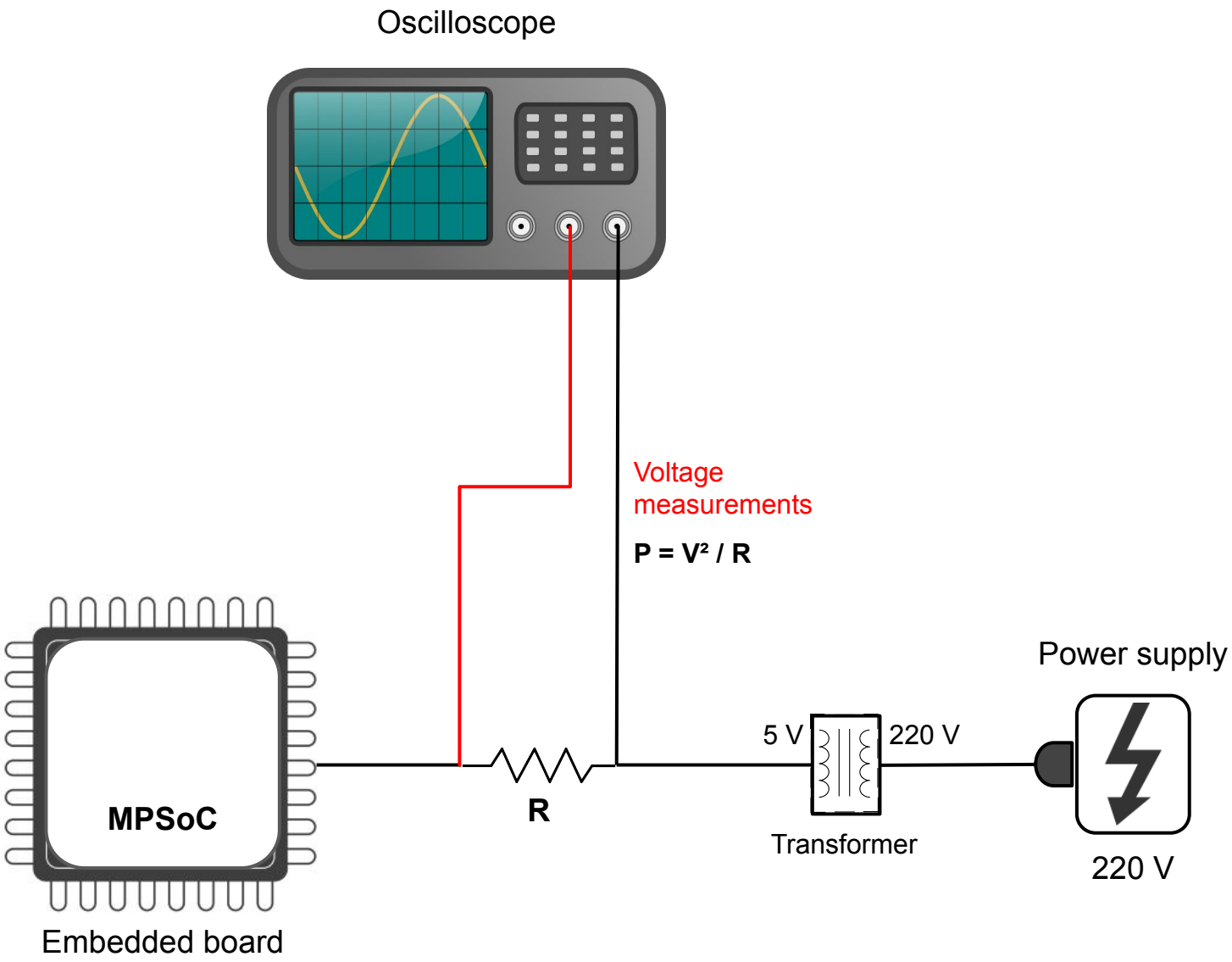
OS & bibliothèques

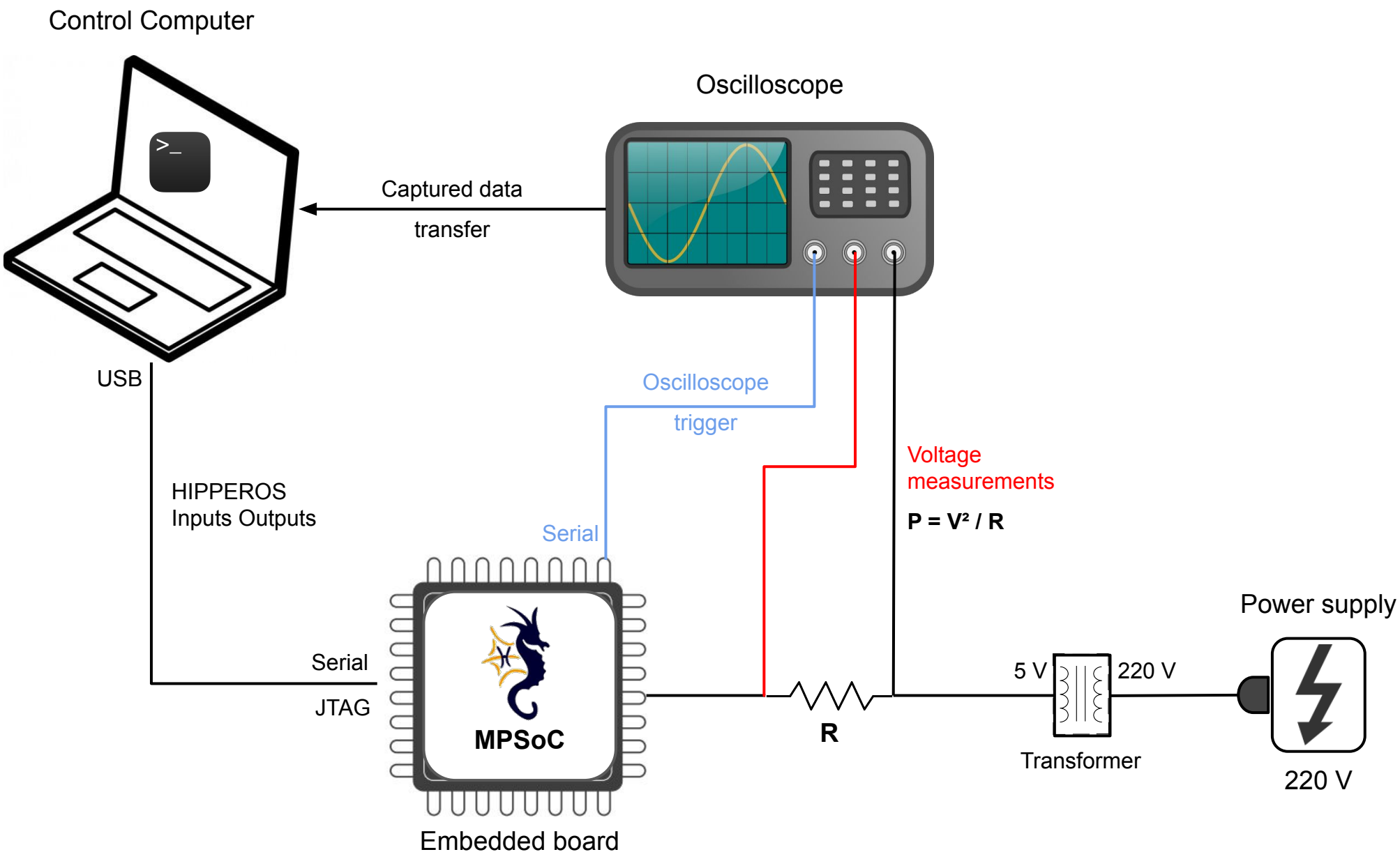
Carte embarquée

Outils de mesure


Power supply
5 V
220 V
Embedded board
Transformer
220 V


MPSoC
Embedded board
5 V
220 V
Transformer
Power supply
220 V

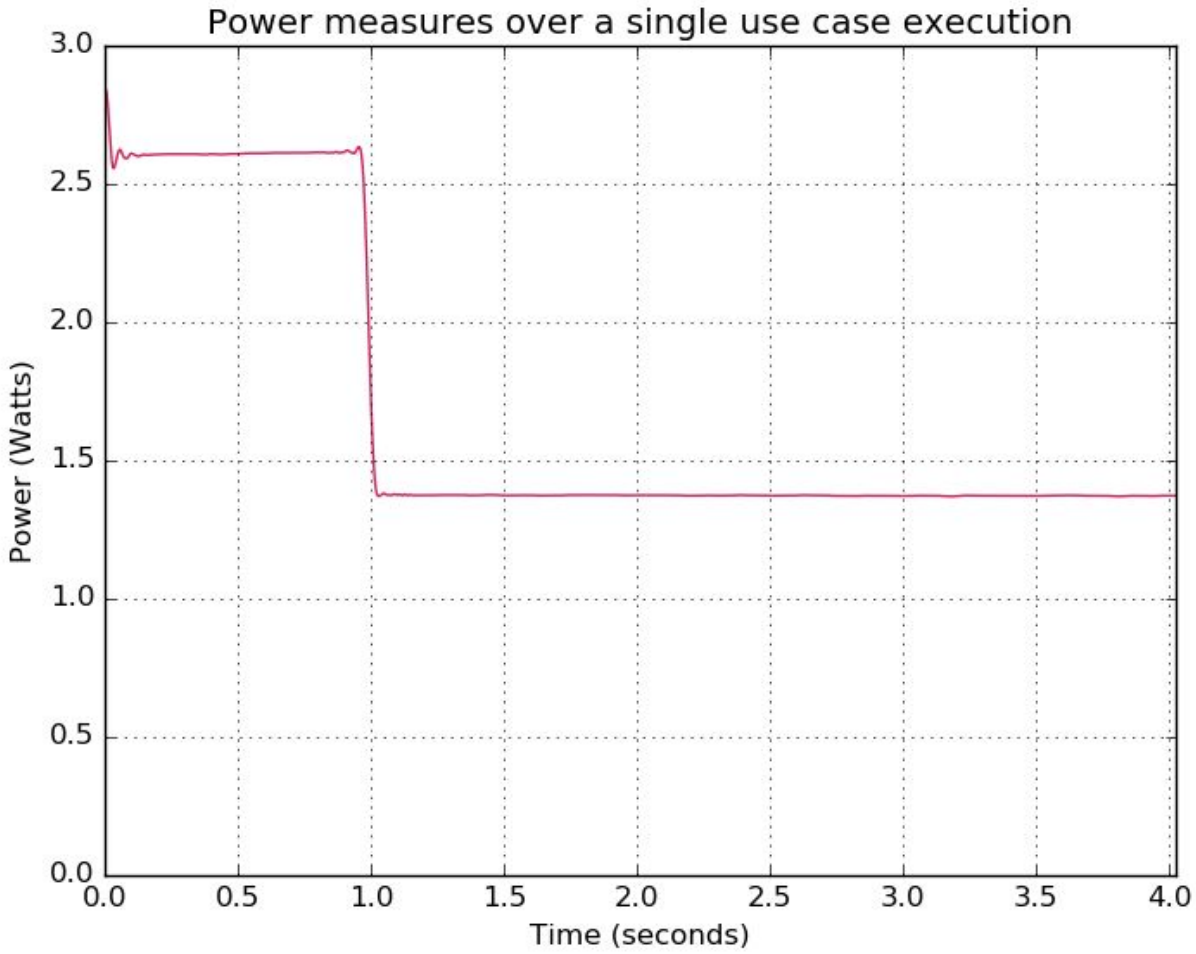

Oscilloscope
Voltage
measurements
P = V² / R
MPSoC
Embedded board
R
5 V
220 V
Transformer
Power supply
220 V


Control Computer
Oscilloscope
Captured data
transfer
USB
Oscilloscope
trigger
Voltage
measurements
P = V² / R
HIPPEROS
Inputs Outputs
Serial
Serial
JTAG
MPSoC
R
5 V
220 V
Transformer
Power supply
220 V
Embedded board

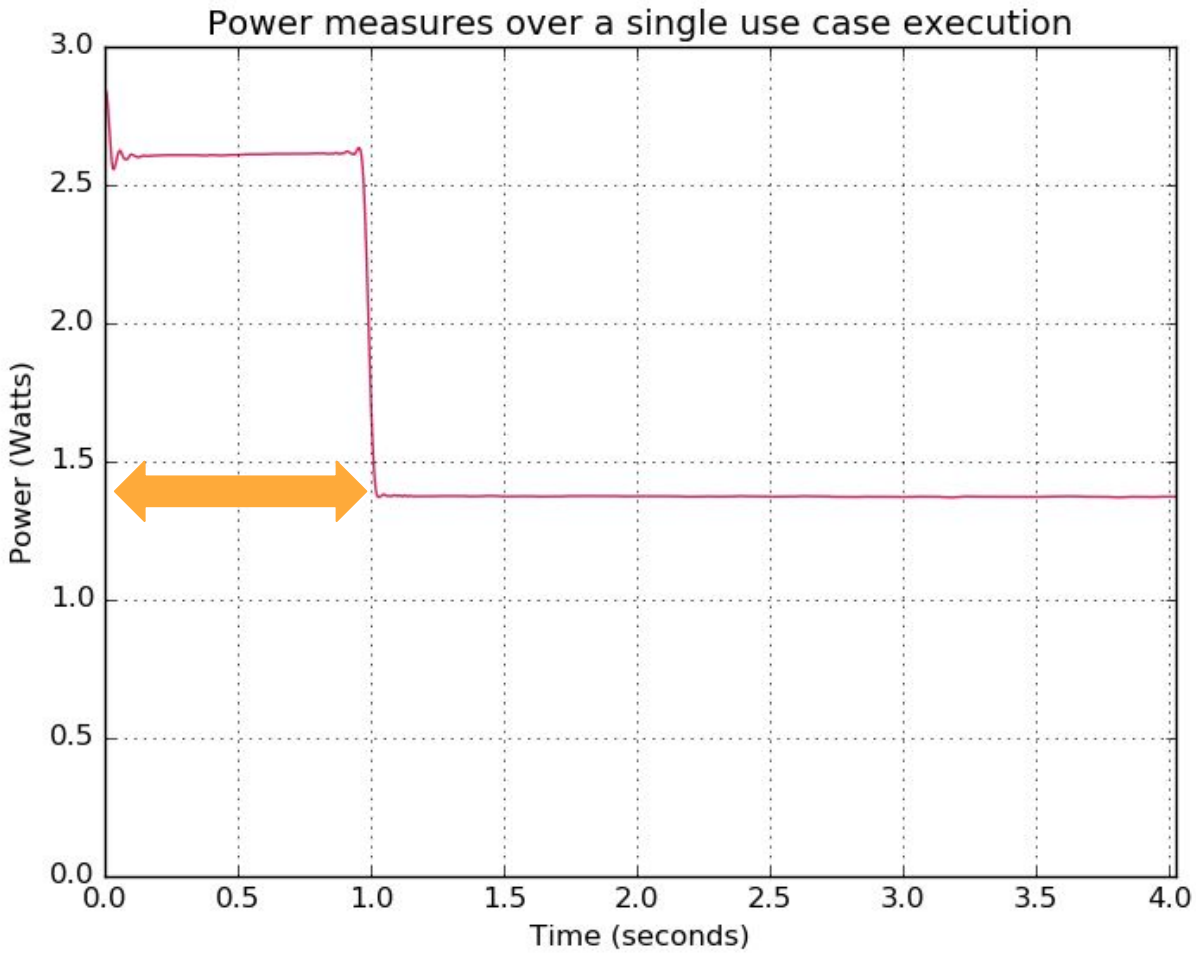

PHILIPS


Keysight Infiniium : Friday, April 08, 2016 5:22:03 AM
KEYSIGHT TECHNOLOGIES
Run
Stop
Single
50.0 Sa/s
High Res
11.0 Hz
T
2.48 V
1 1MΩ 140 mV/
444 mV
2 1MΩ 1.70 V/
-900 mV
Time Meas
Vertical Meas
Measurements
1.00 V
864 mV
724 mV
584 mV
444 mV
304 mV
164 mV
24 mV
-116 mV
-999 ms
-499 ms
1 ms
501 ms
1.00 s
1.50 s
2.00 s
2.50 s
3.00 s
3.50 s
4.00 s
H
500 ms/
1.6012800000000 s
Results

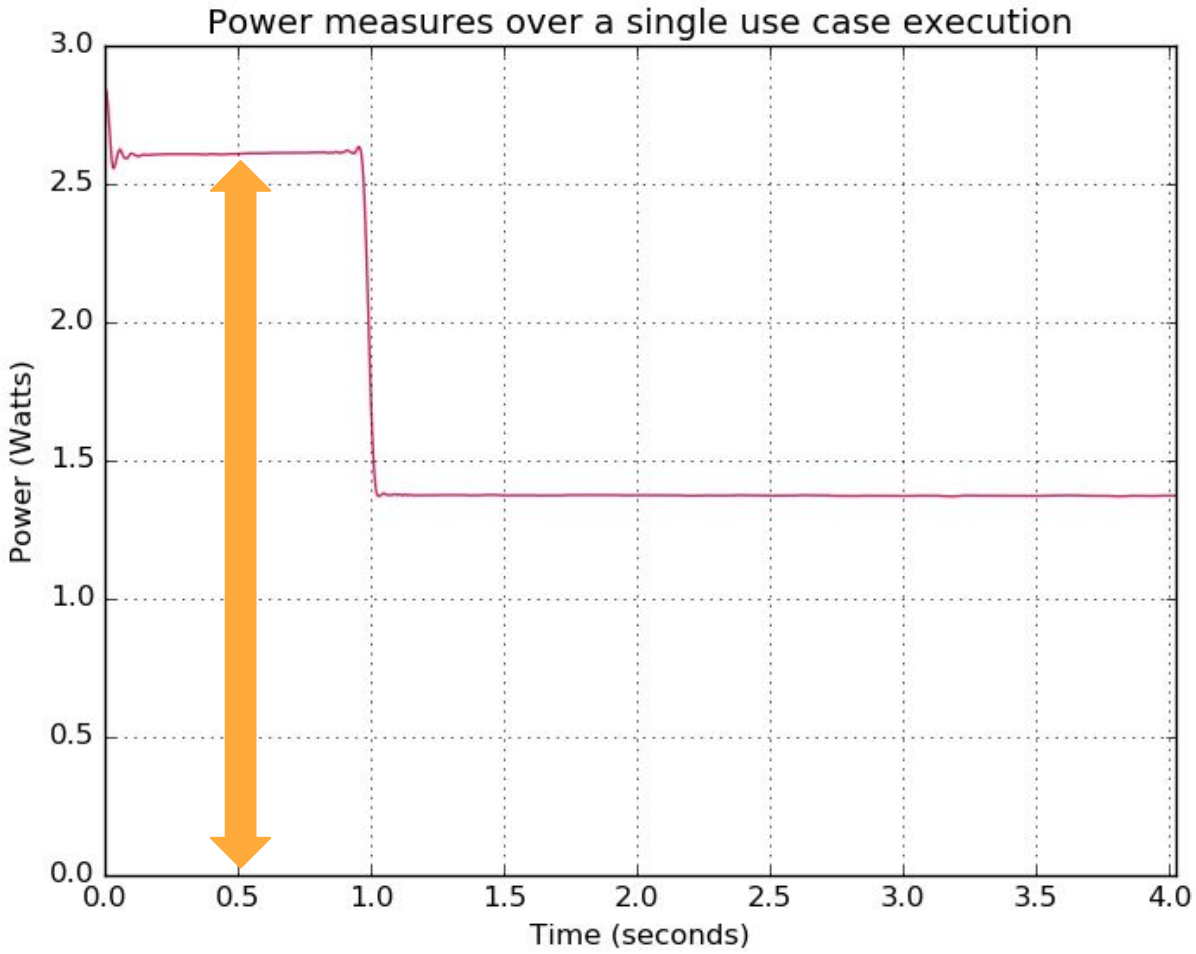
Voltage converti en puissance / temps



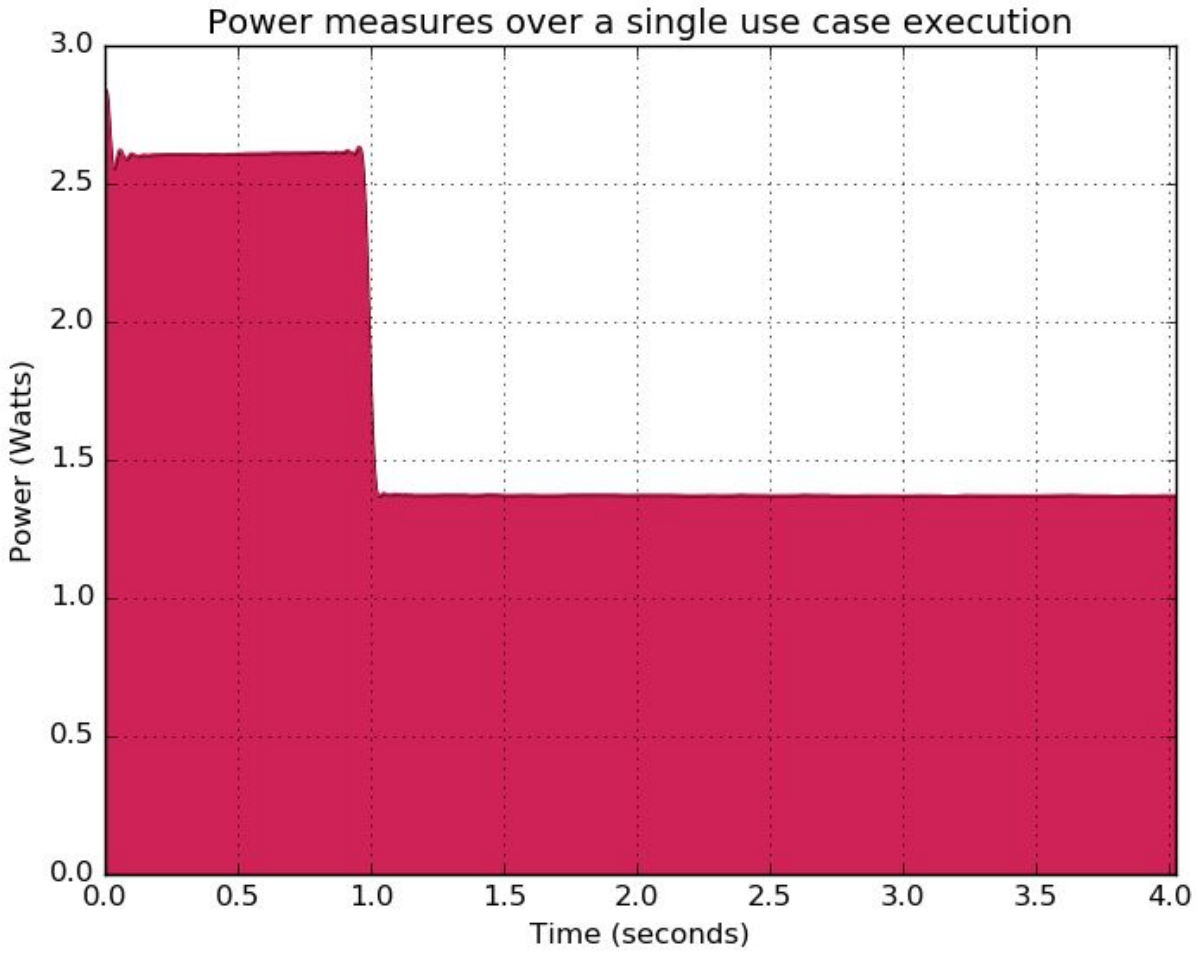
Temps d’exécution



Pic de puissance

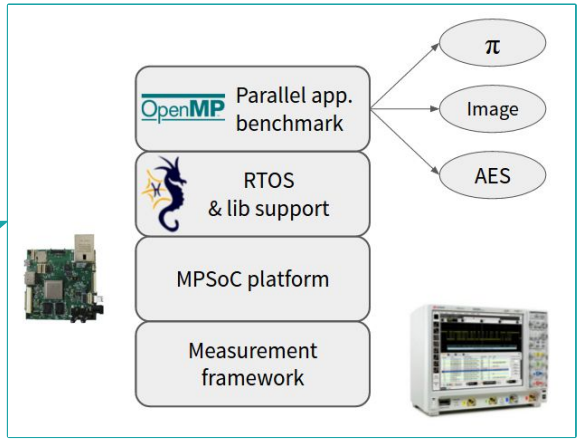


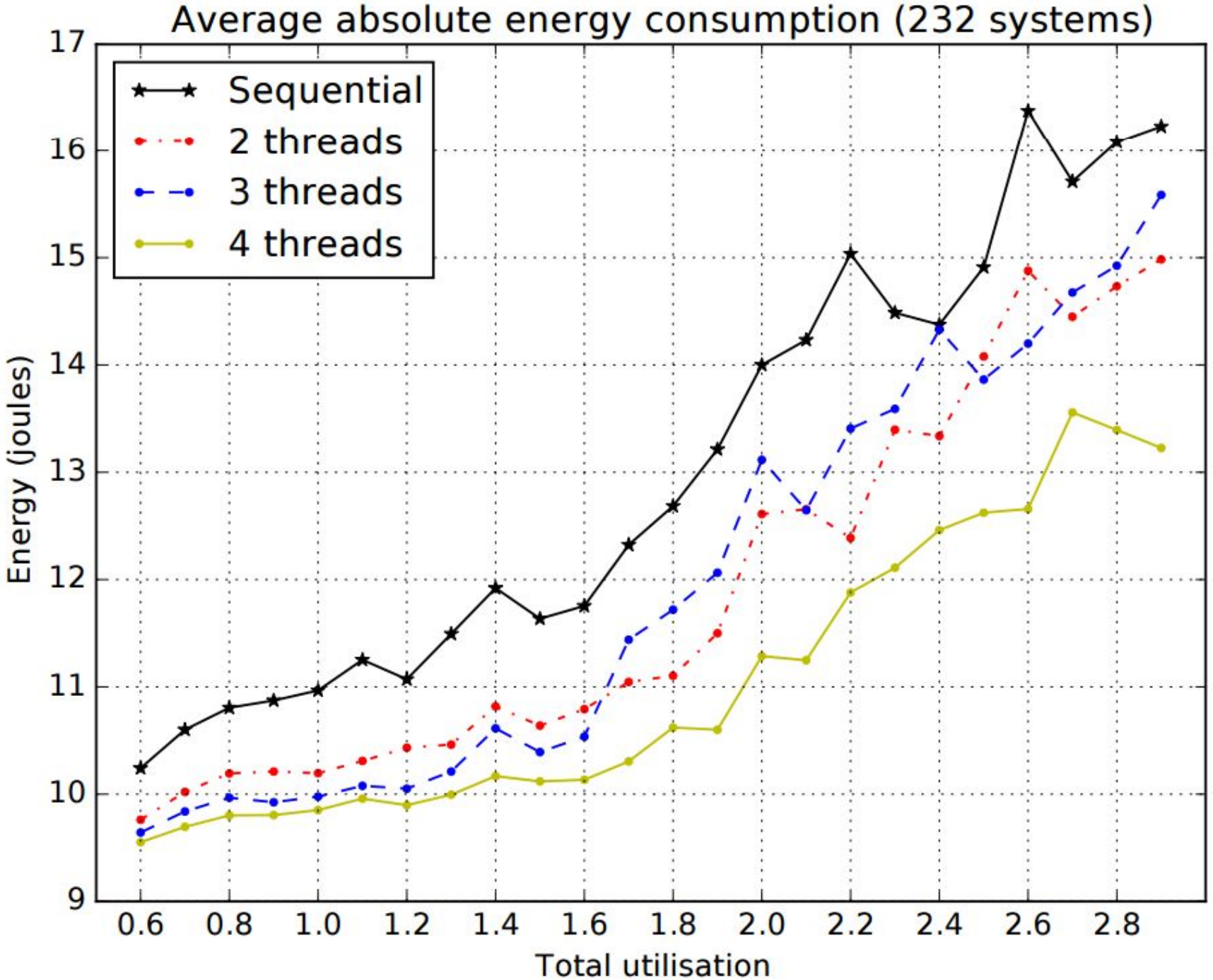
Energie consommée

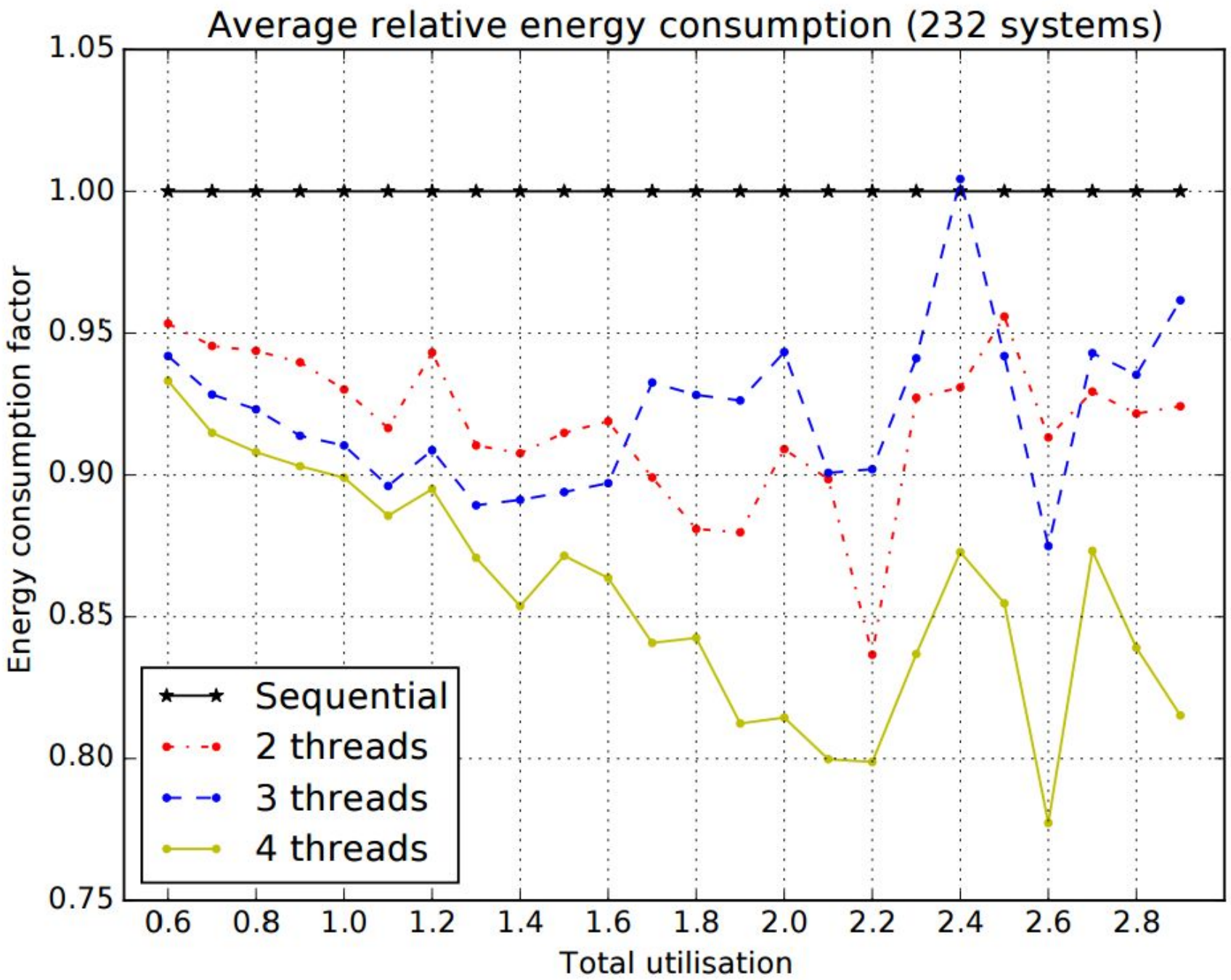


Campagne de simulations expérimentales

- Génération aléatoire de tâches applicatives à charge variable (U = 0.6 → 2.9)
- Pour différents degrés de parallélisme - **1, 2, 3, 4 threads**
- L'OS applique la gestion de l'alimentation
- On exécute les systèmes ordonnançables
- On mesure les résultats




Average absolute energy consumption (232 systems)
Sequential
2 threads
3 threads
4 threads
Energy (joules)
Total utilisation
17
16
15
14
13
12
11
10
9
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
2.2
2.4
2.6
2.8


Average relative energy consumption (232 systems)
Energy consumption factor
1.05
1.00
0.95
0.90
0.85
0.80
0.75
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
2.2
2.4
2.6
2.8
Total utilisation
Sequential
2 threads
3 threads
4 threads

Conclusion

Le parallélisme permet de
réduire l'énergie consommée tout en
garantissant **les contraintes temporelles**

Agenda

1. Domaine, problème, intuition
2. Contributions et thèse
3. **Directions futures**

`len(`**Ph.D**`)`

=

4 ans

`len(`**Ph.D**`)`

≃

5.5 ans

`len(`**Ph.D**`)`

≃

5.5 ans

→

Vision
Haute Altitude

Conception systèmes embarqués, **aujourd'hui**

```
do {
    useful_work();
}
```

Communauté temps-réel, conception, **vision**

```
do in max 3 ms {
    useful_work();
}
```

Conception systèmes embarqués, **ma vision**

```
do in max 3 ms && in max 3 mJ {
    useful_work();
}
```

Ma vision, **directions futures**

```
do in max (3 ms,3 mJ,2°C) {
    useful_work();
}
```

Réaction du système

```
do in max (3 ms,3 mJ,2°C) {
    useful_work();
}
```

```
$ cc useful.c
```

Réaction du système

```
do in max (3 ms,3 mJ,2°C) {
    useful_work();
}
```

```
$ cc useful.c
.
```

Réaction du système

```
do in max (3 ms,3 mJ,2°C) {
    useful_work();
}
```

```
$ cc useful.c
..
```

Réaction du système

```
do in max (3 ms,3 mJ,2°C) {
    useful_work();
}
```

```
$ cc useful.c
OK (ordonnançable)
$
```

System reaction - resource requirements

```
do in max (3 ms,3 mJ,2°C) {
    useful_work();
}
```

```
$ cc useful.c
Erreur - veuillez allouer 2 cœurs à la fonction
‘useful_work’ (non ordonnançable).
$
```

Merci.

Q ?

...

• • • •

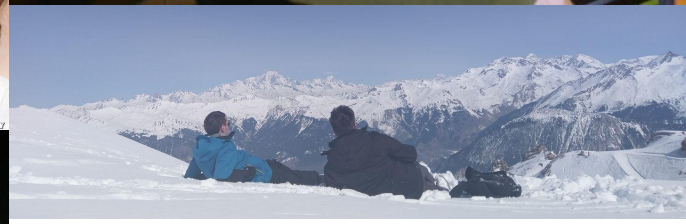
• • • • •

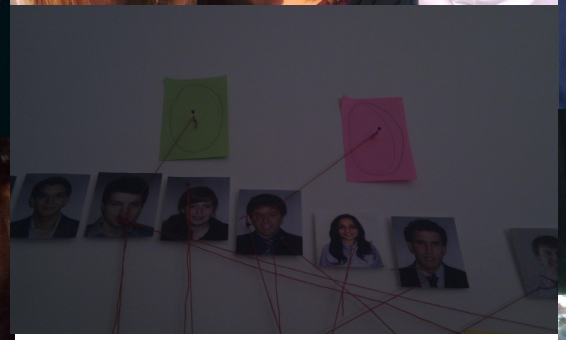
VICKINGS

FEAR THEM

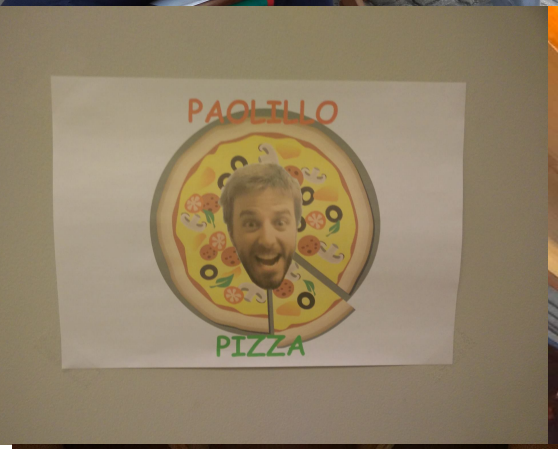

SUPREME
SINCE 1968
QUALITY
CRAFTED WITH PASSION
GOD SAVE THE 90'S


Aria & Bruno

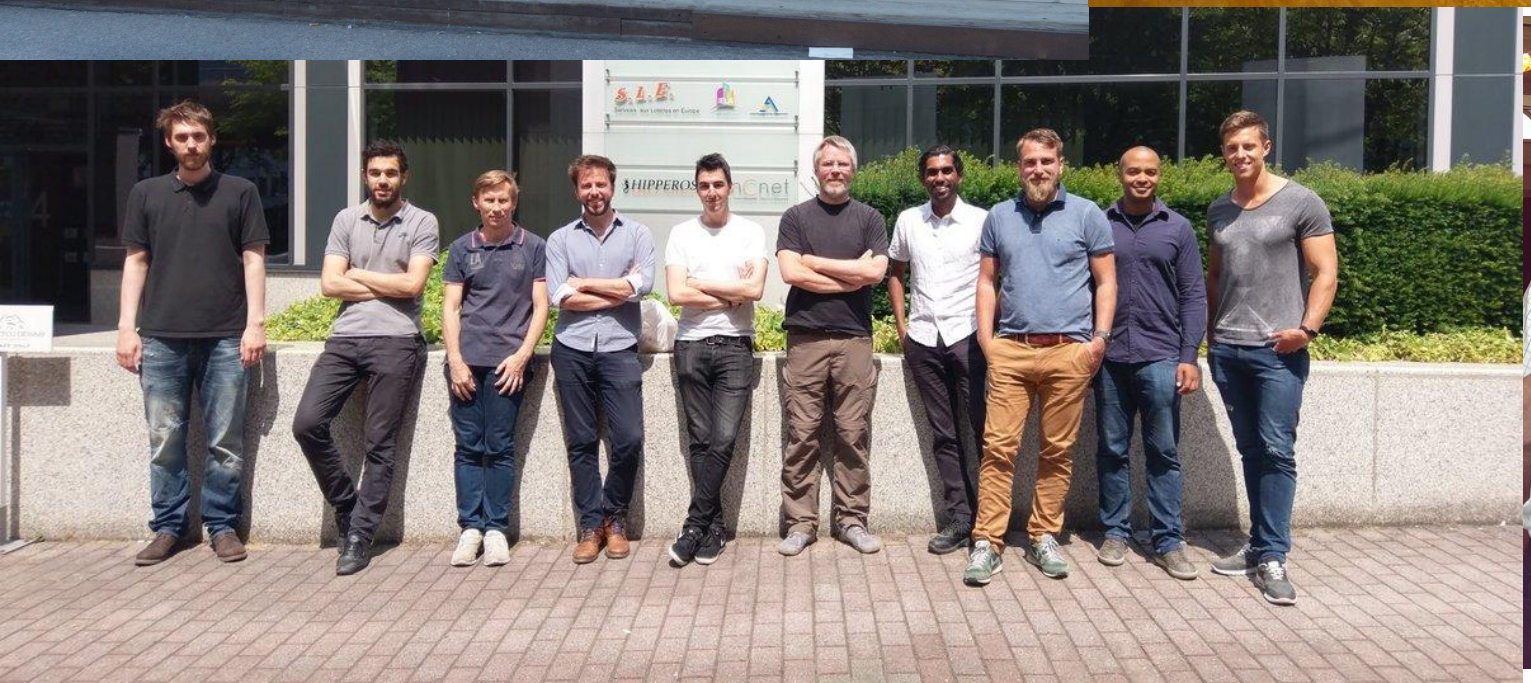


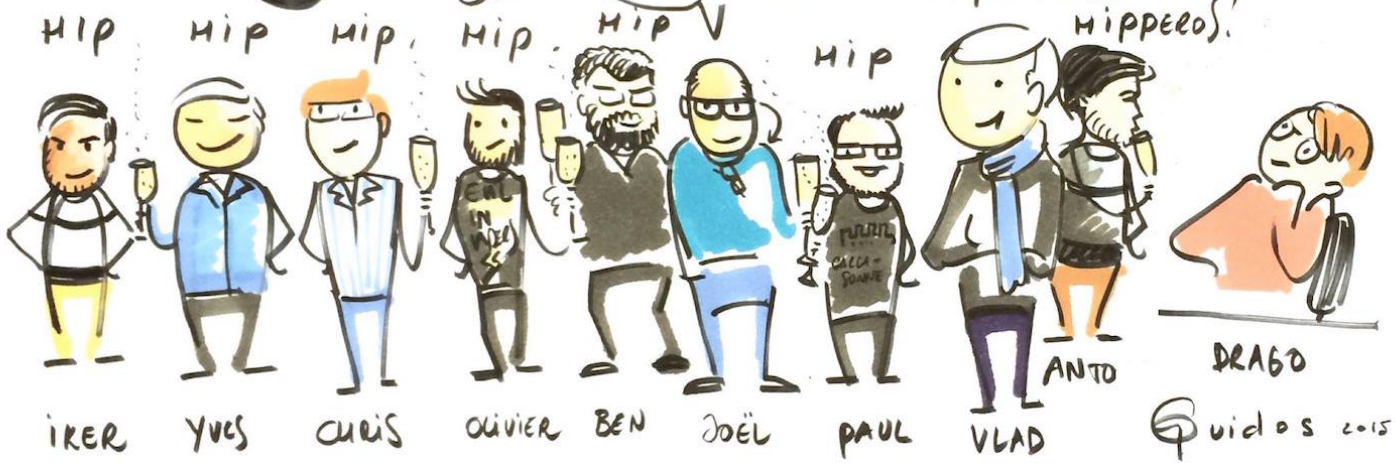



MONUMENT

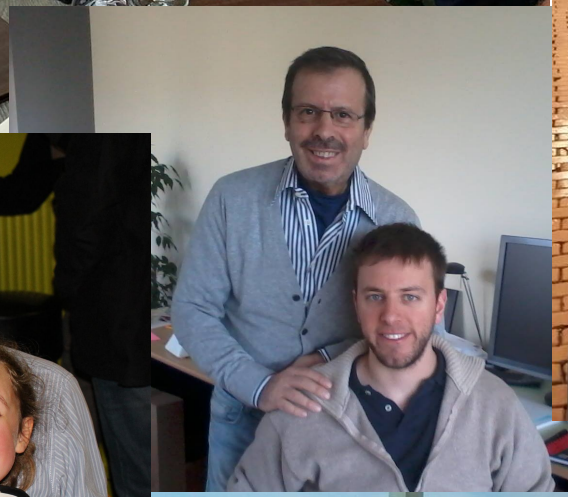

PAOLILLO
PIZZA


Your Partner in
High Performance
Design




HIP
HIP
HIP
HIP
HIP
HIP
HIPPEOS!
IKER
YVES
CHRIS
OLIVIER
BEN
JOËL
PAUL
VLAD
ANTO
DRAGO
Guidos 2015


HILFIGER
AMERICA FINEST
SURFING
DENIM CO.




SURFING


Your journal has been updated


Merci.

Merci.


THESIS STARTED
imgflip.com
THESIS DEFENDED