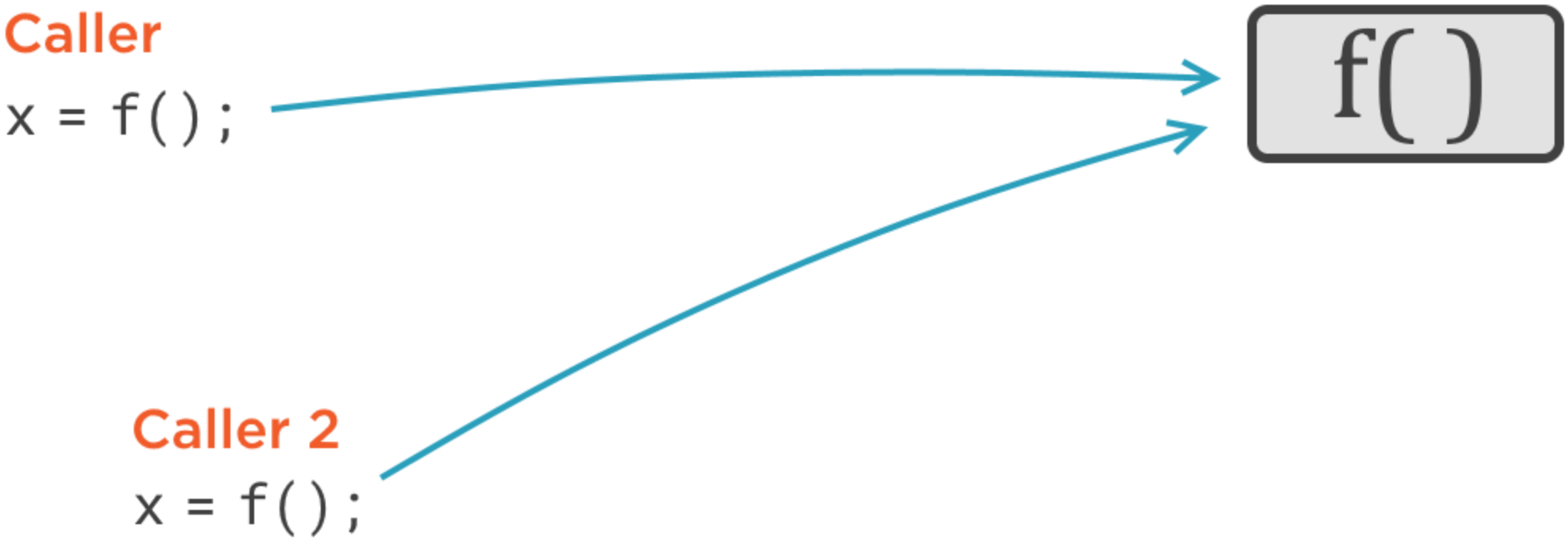
Memoization with Pure Functions

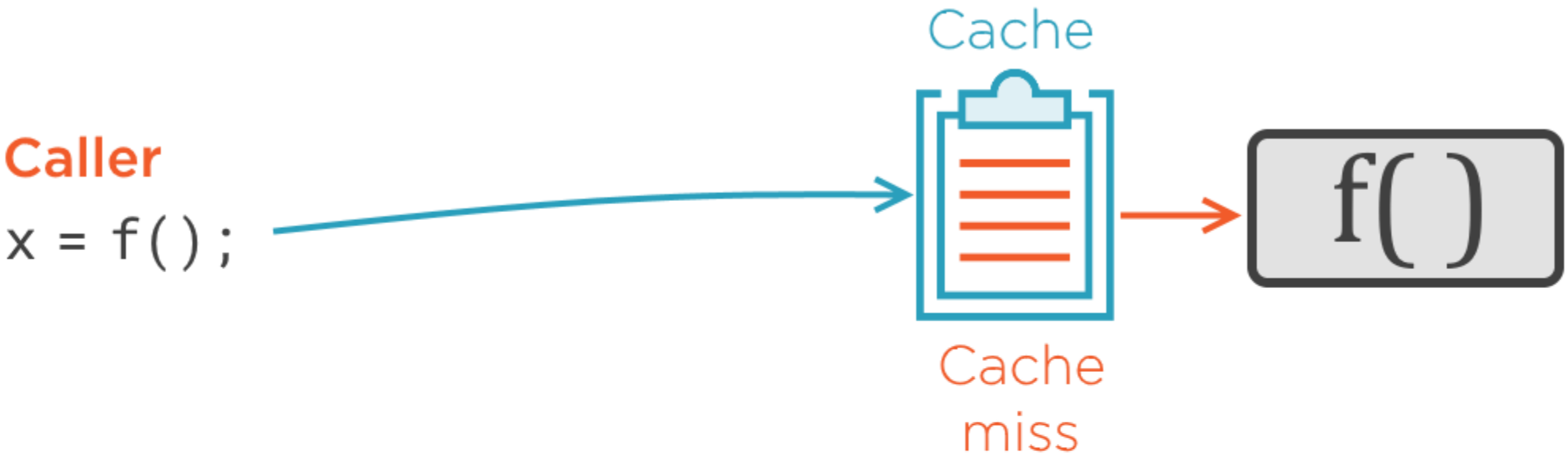
Zoran Horvat
CEO AT CODING HELMET

@zoranh75 http://csharpmentor.com

Memoization as a Trade-off



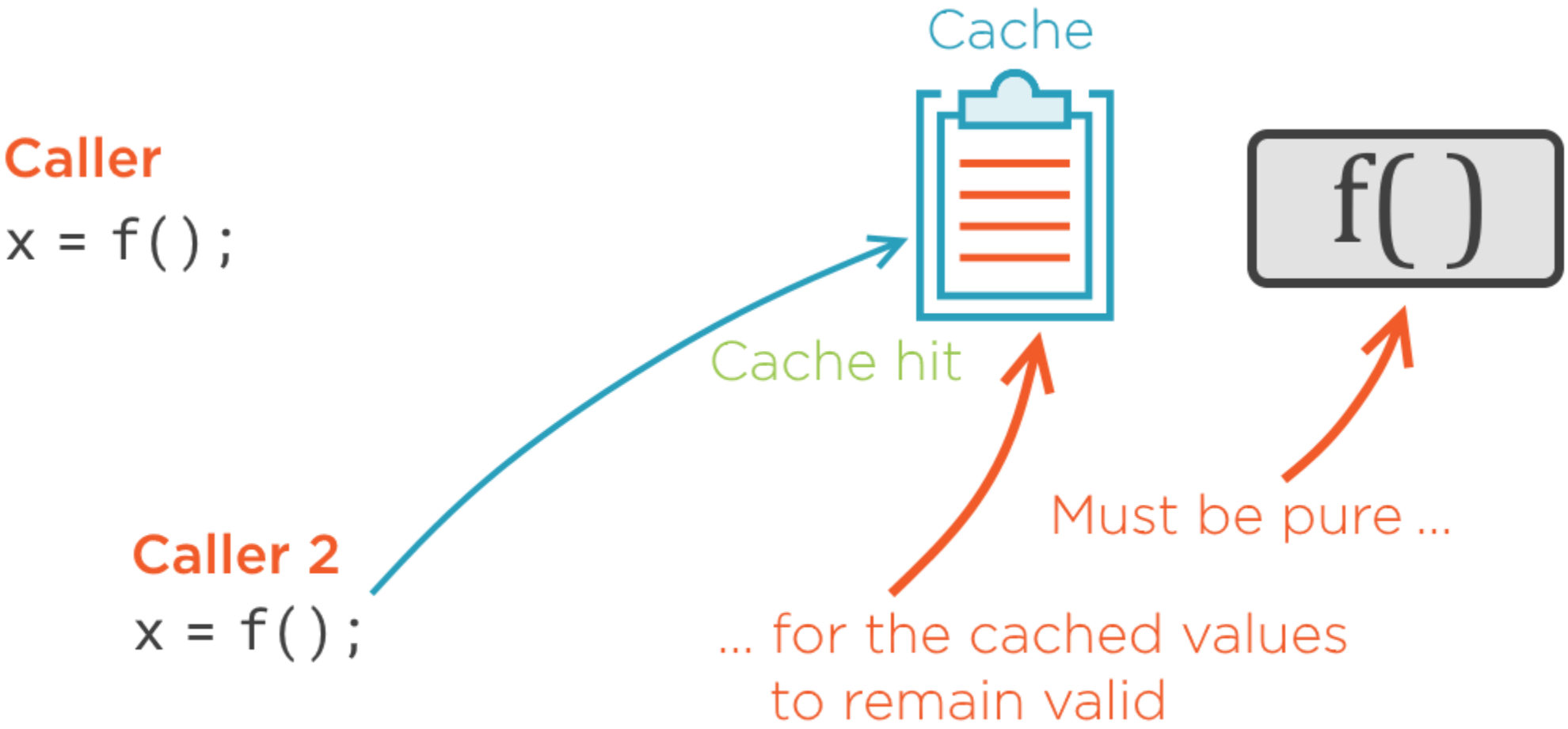
Memoization as a Trade-off



Caller 2

```
x = f();
```

Memoization as a Trade-off



Memoization as a Trade-off

Caller

```
x = f();
```

Cache

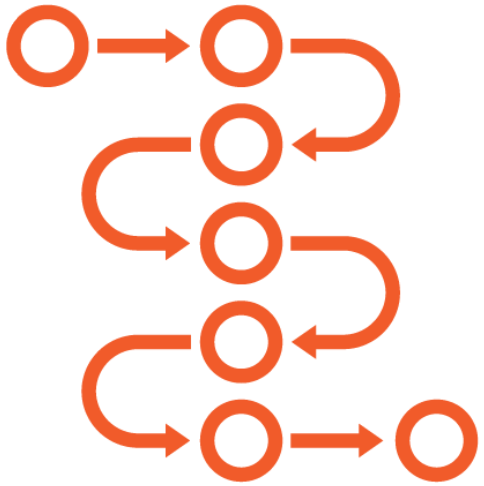
Larger memory footprint
Possible defects

Caller 2

```
x = f();
```

You have a problem.
You choose to solve it by caching results.
Now you have two problems.

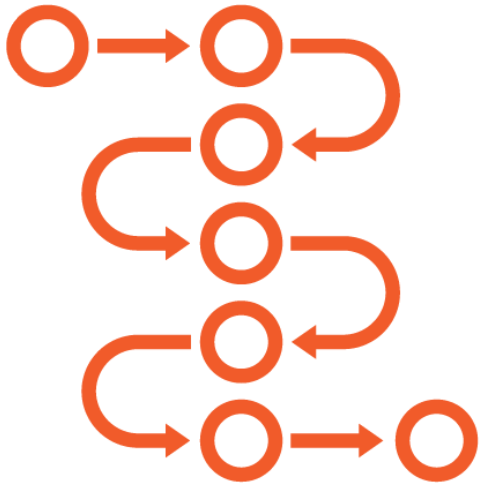
Dynamic Programming Defined



Dynamic programming (a.k.a. dynamic optimization) is a method for solving a complex problem by breaking it down into a collection of simpler subproblems, solving each of those subproblems just once, and storing their solutions.

https://en.wikipedia.org/wiki/Dynamic_programming

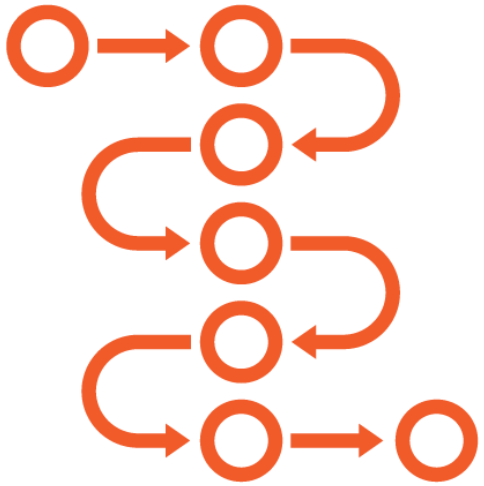
Dynamic Programming Defined

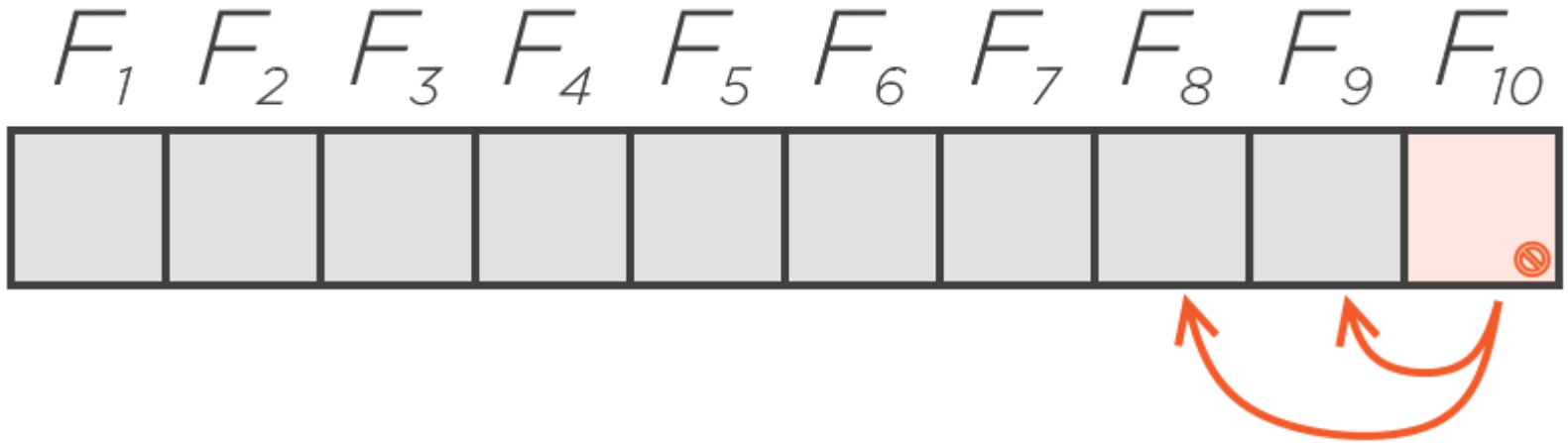


Dynamic programming (a.k.a. dynamic optimization) is a method for solving a **complex problem** by breaking it down into a collection of simpler **subproblems**, solving each of those subproblems just **once**, and **storing** their solutions.

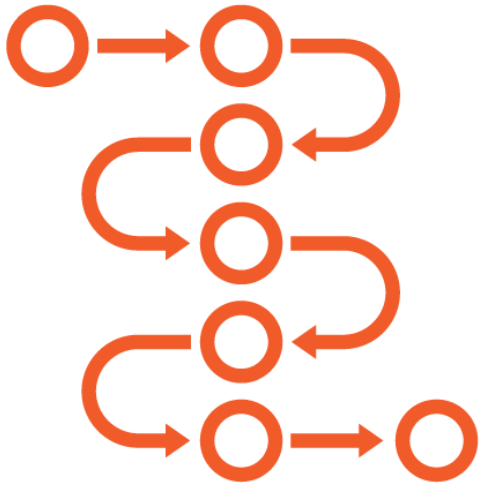
https://en.wikipedia.org/wiki/Dynamic_programming

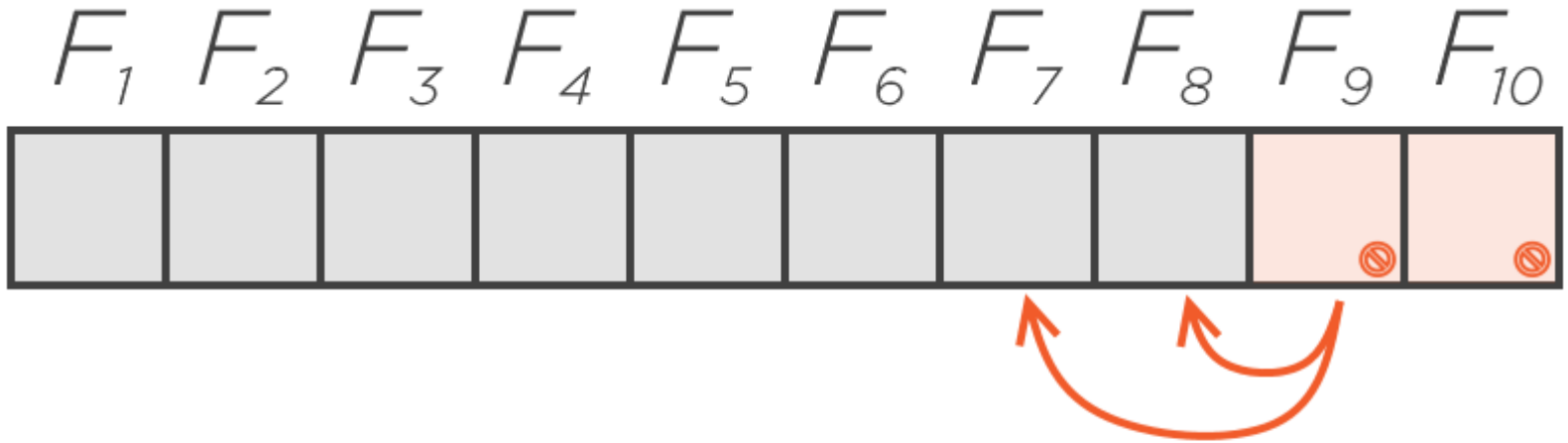
Dynamic Programming Defined



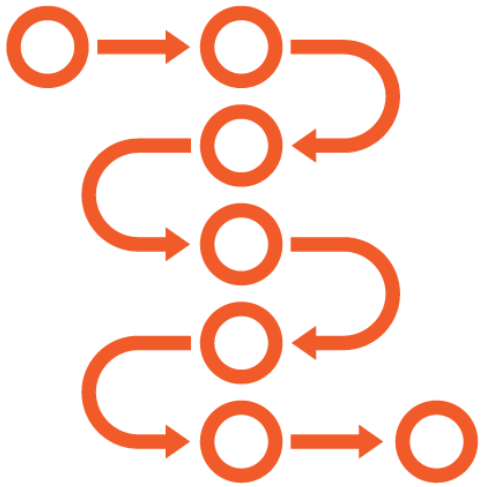


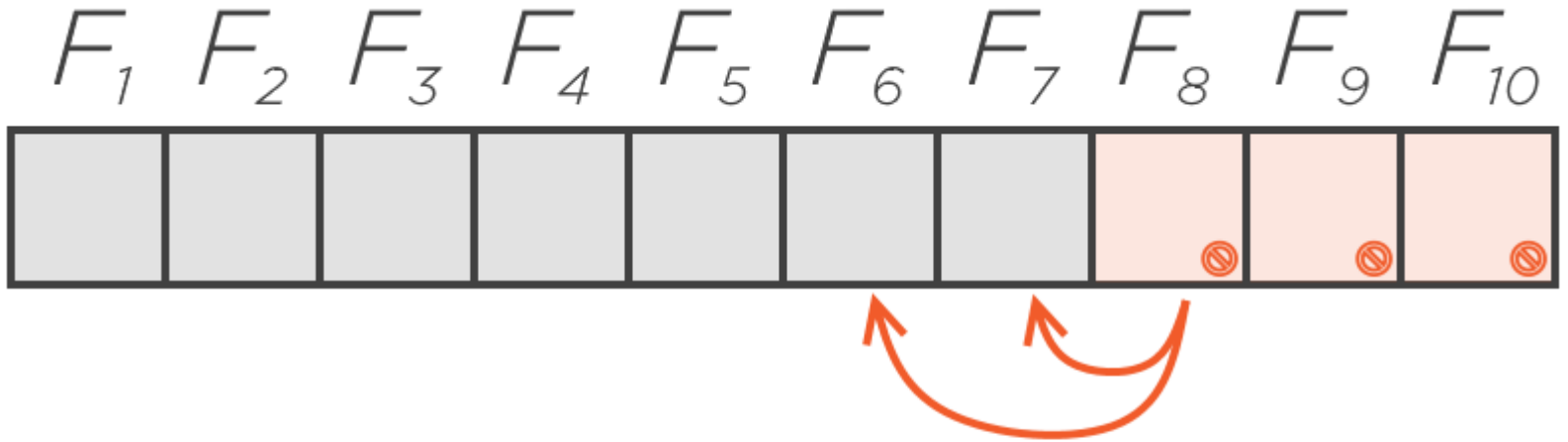
Dynamic Programming Defined



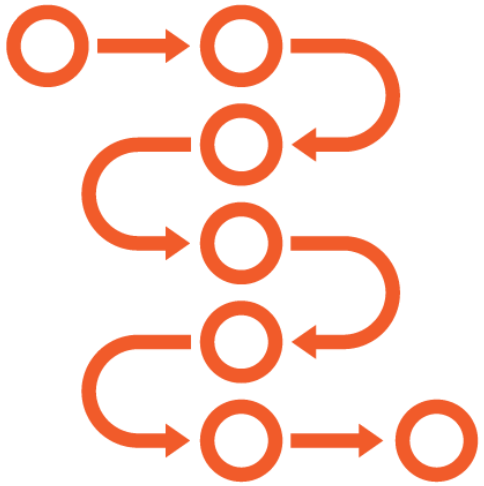


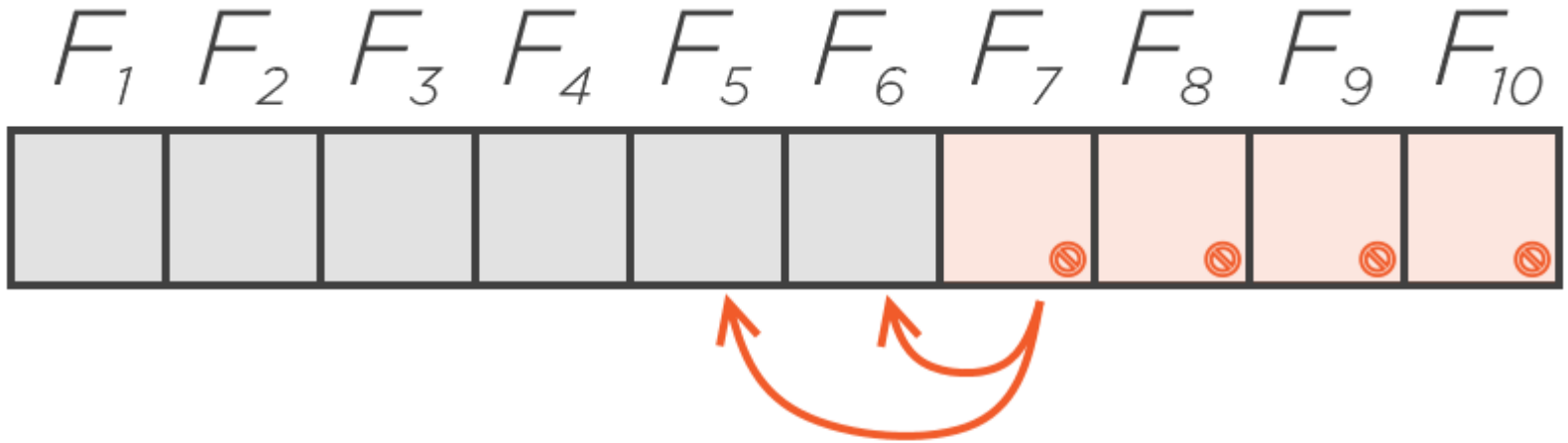
Dynamic Programming Defined



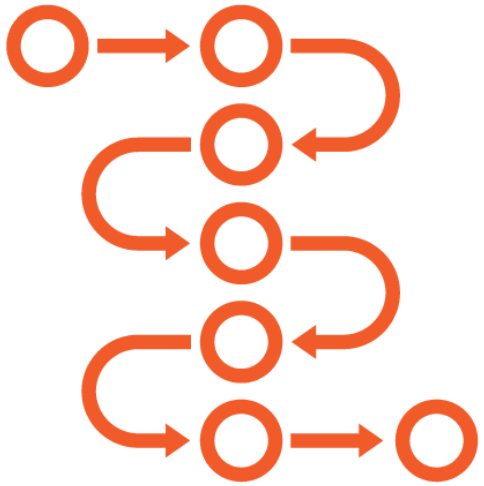


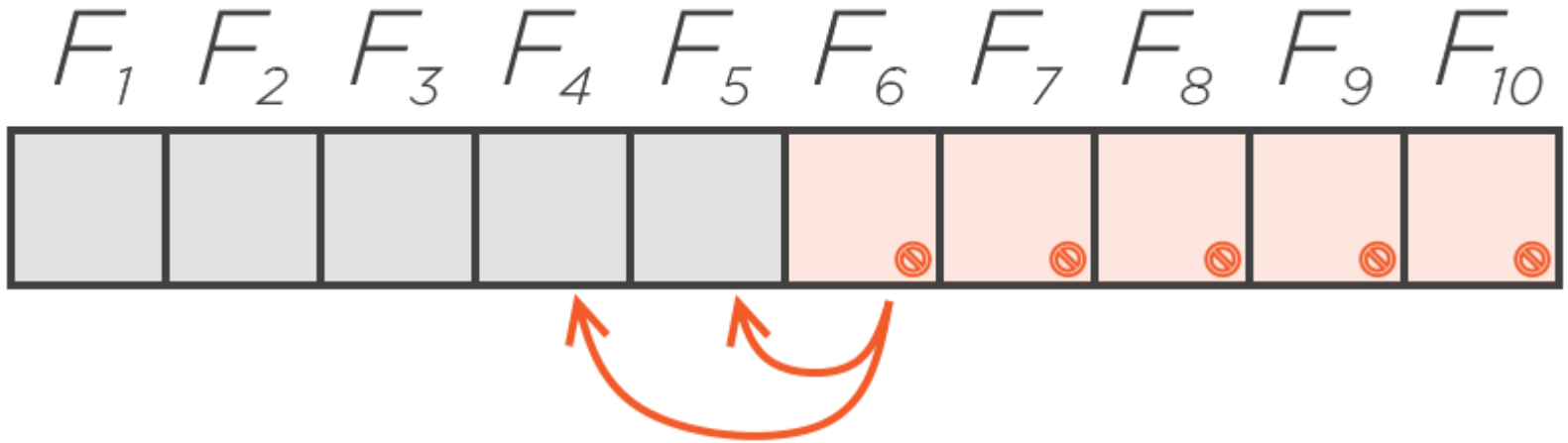
Dynamic Programming Defined



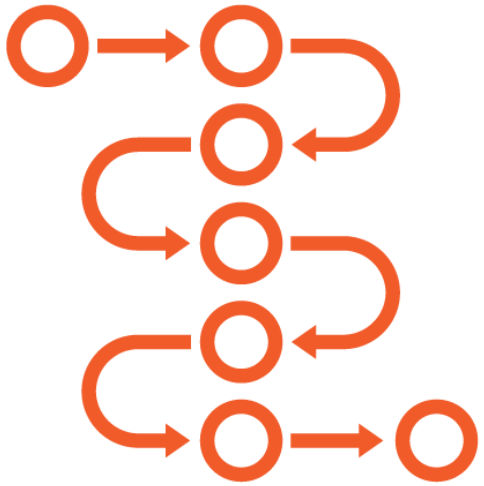


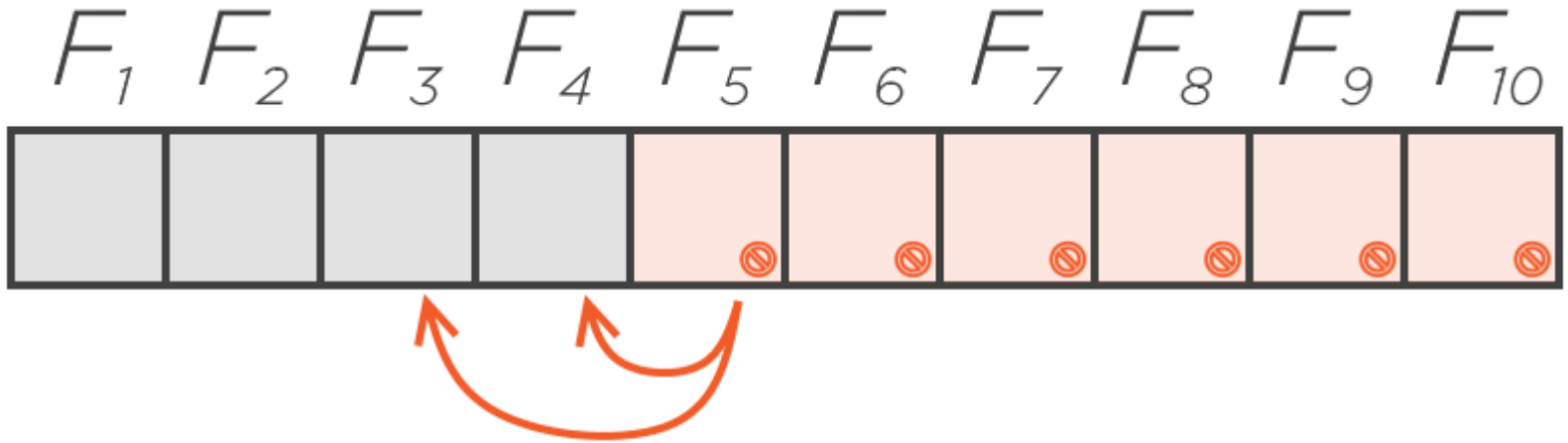
Dynamic Programming Defined



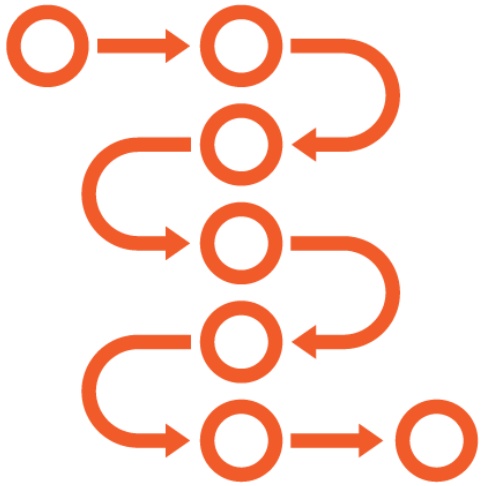


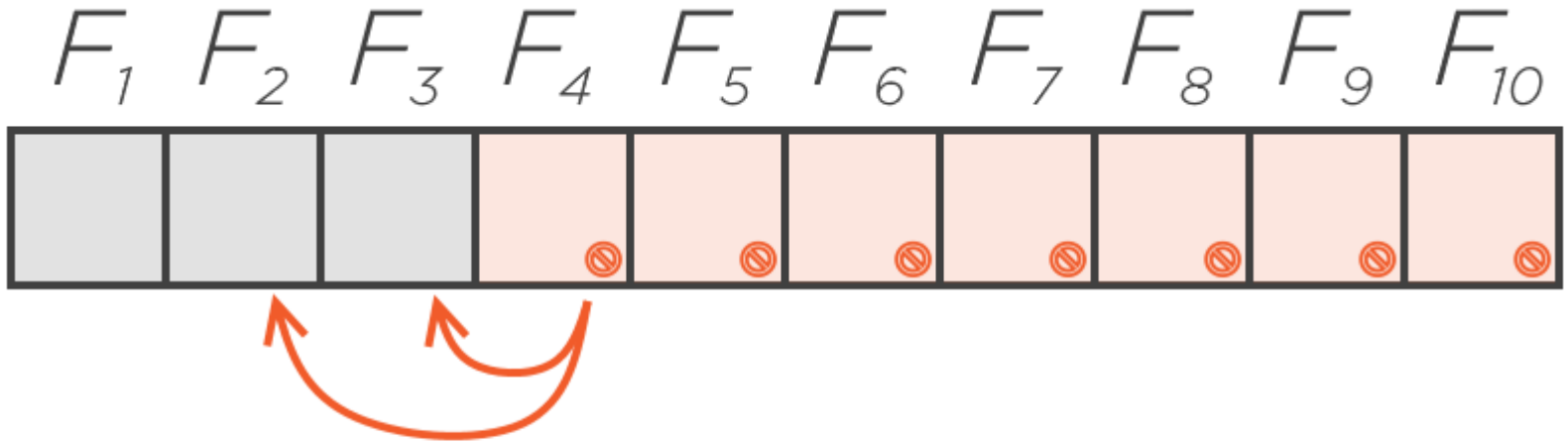
Dynamic Programming Defined



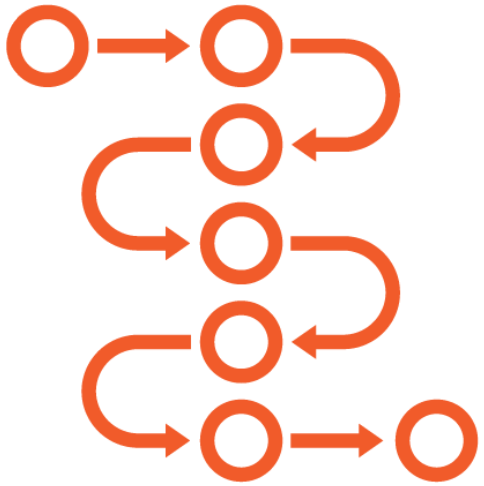


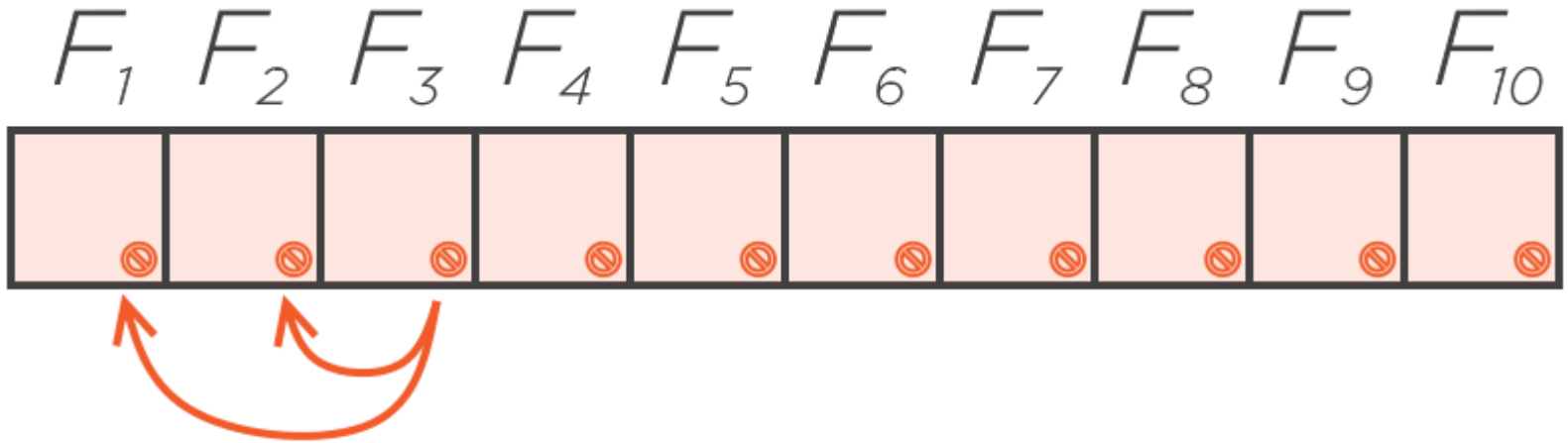
Dynamic Programming Defined



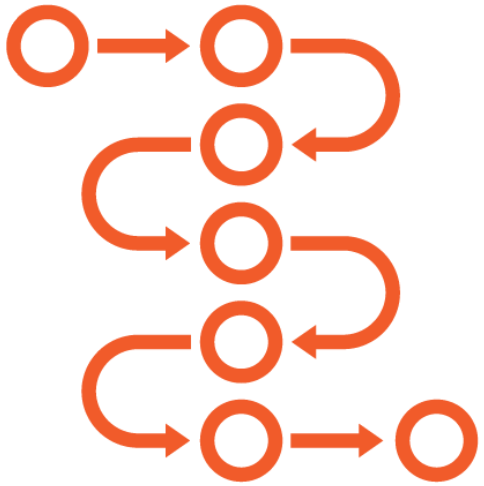


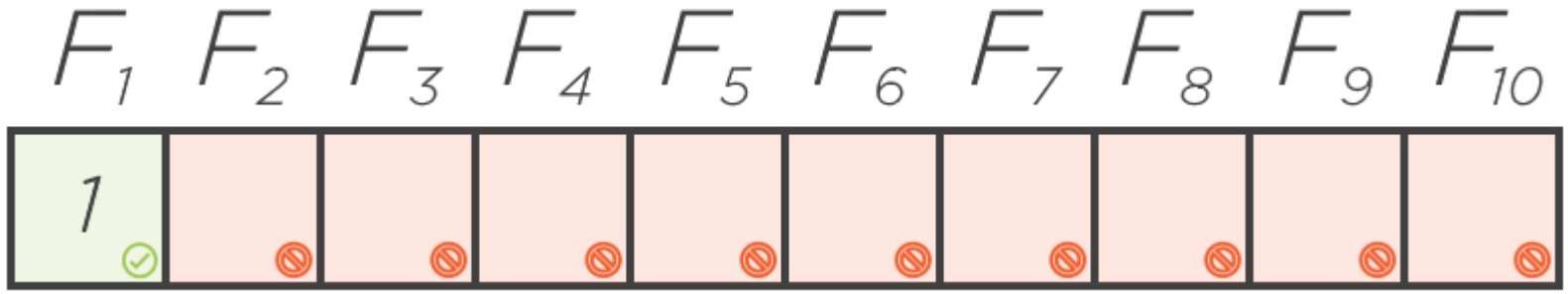
Dynamic Programming Defined



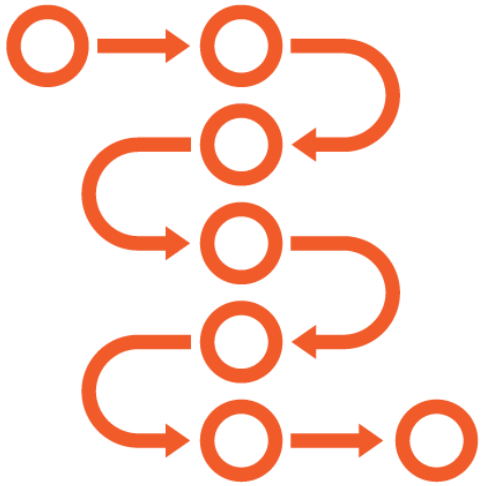


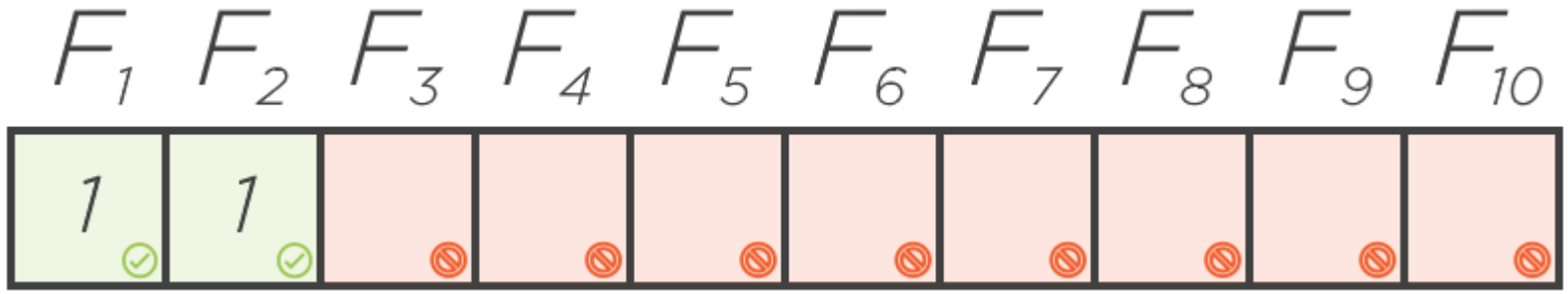
Dynamic Programming Defined



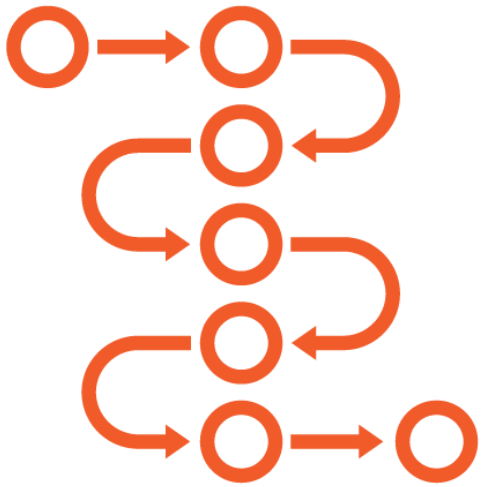


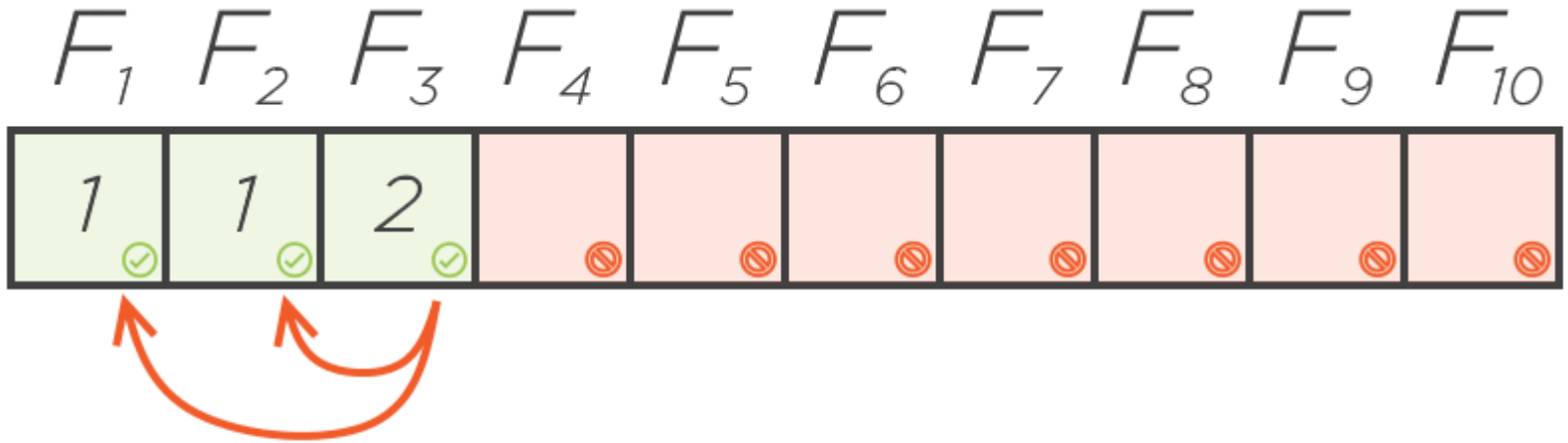
Dynamic Programming Defined



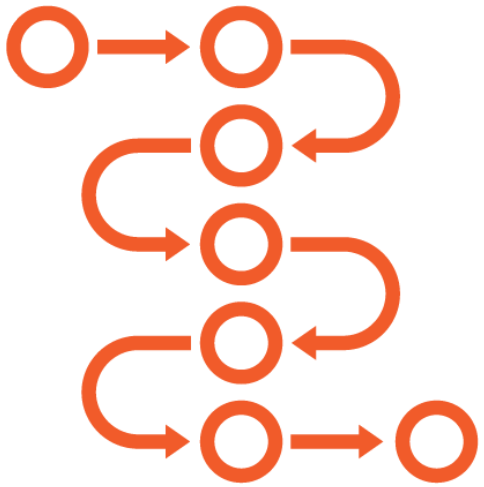


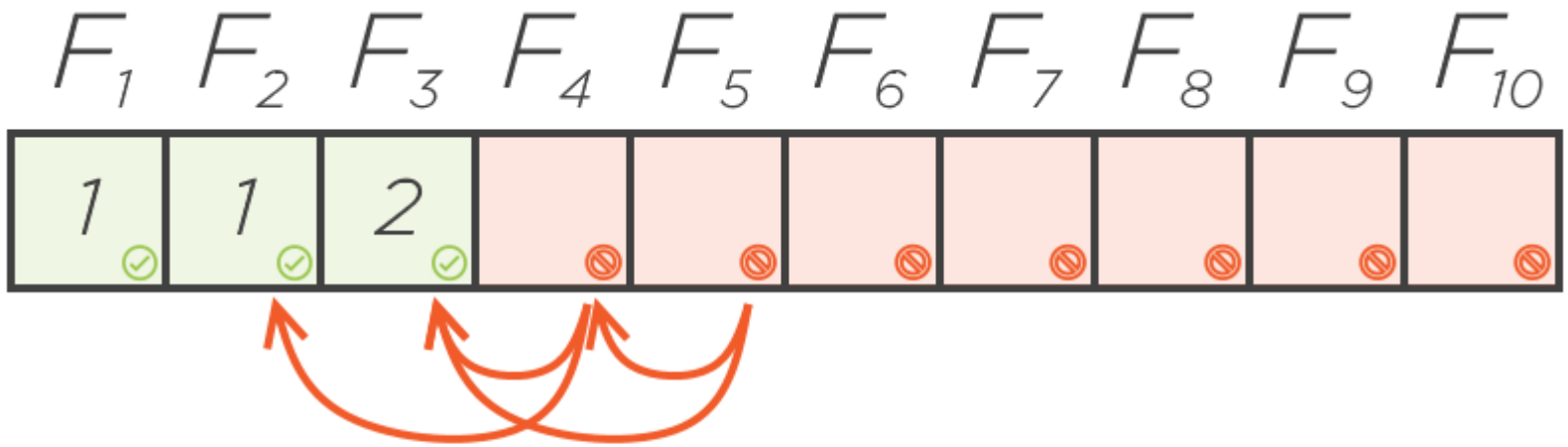
Dynamic Programming Defined



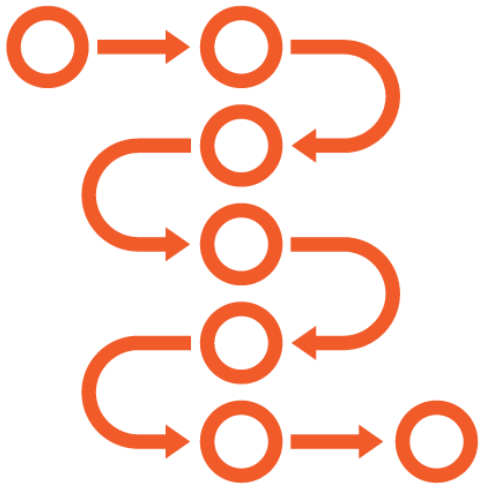


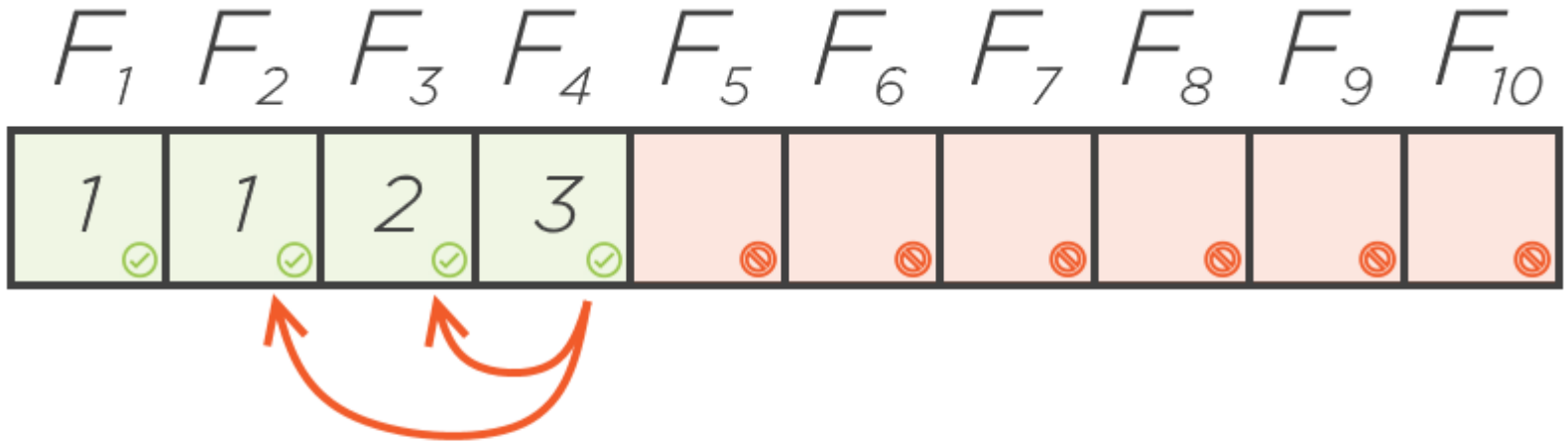
Dynamic Programming Defined



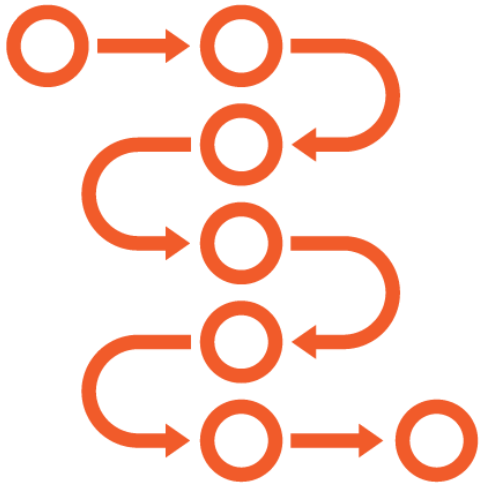


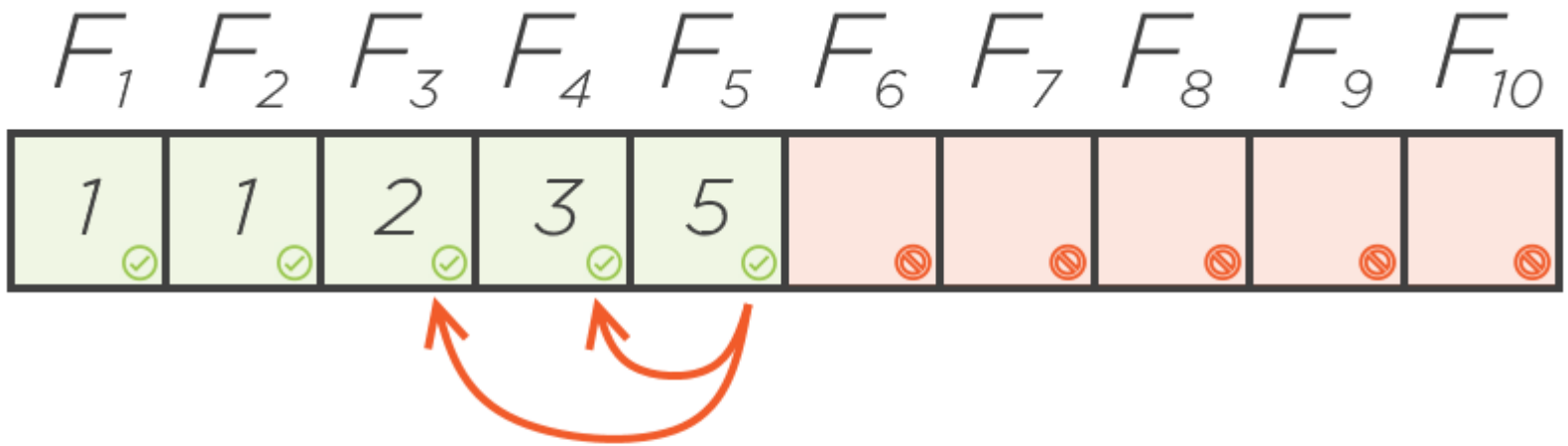
Dynamic Programming Defined



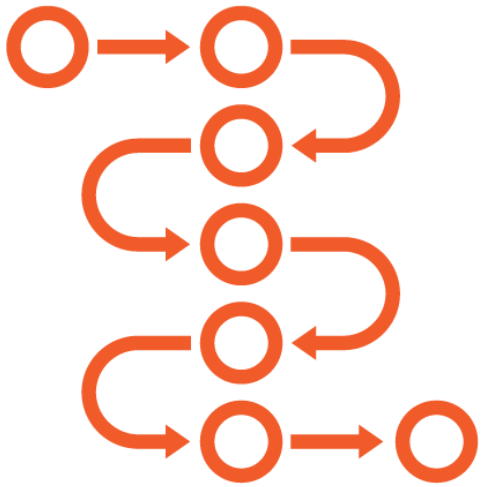


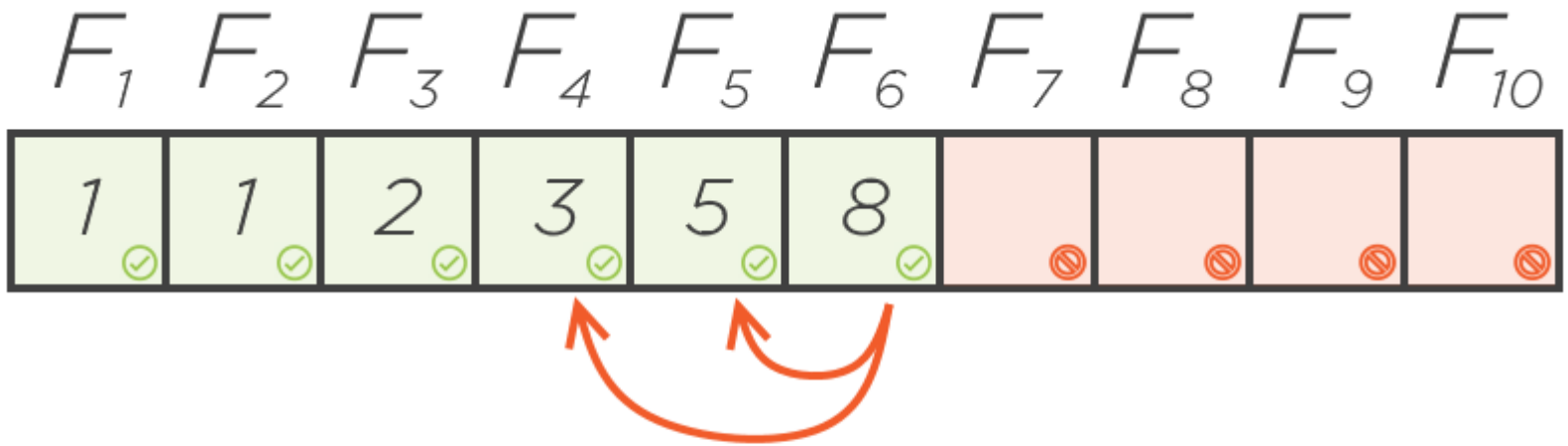
Dynamic Programming Defined



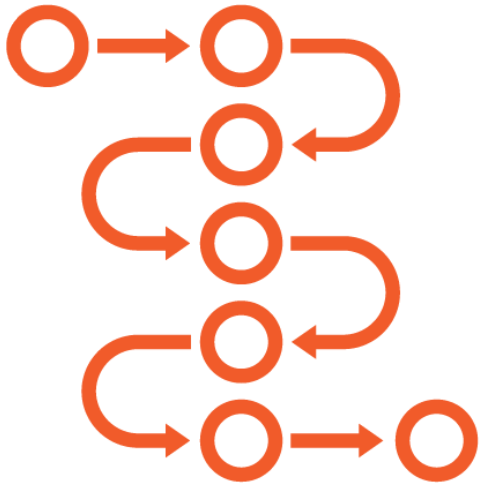


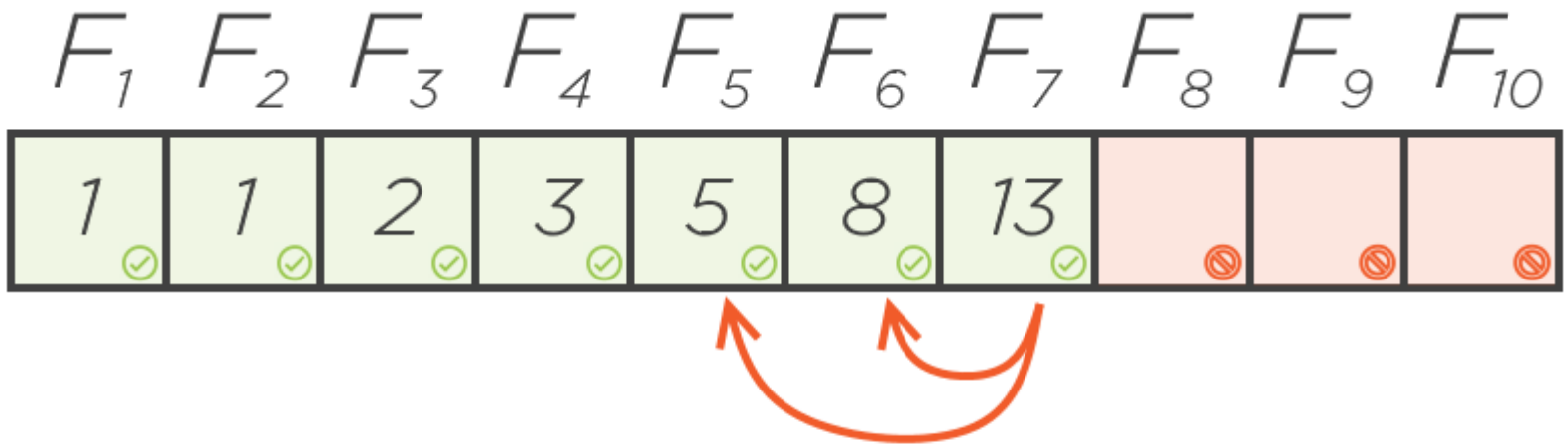
Dynamic Programming Defined



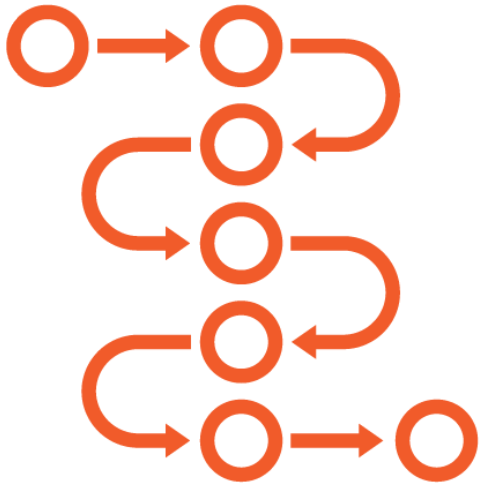


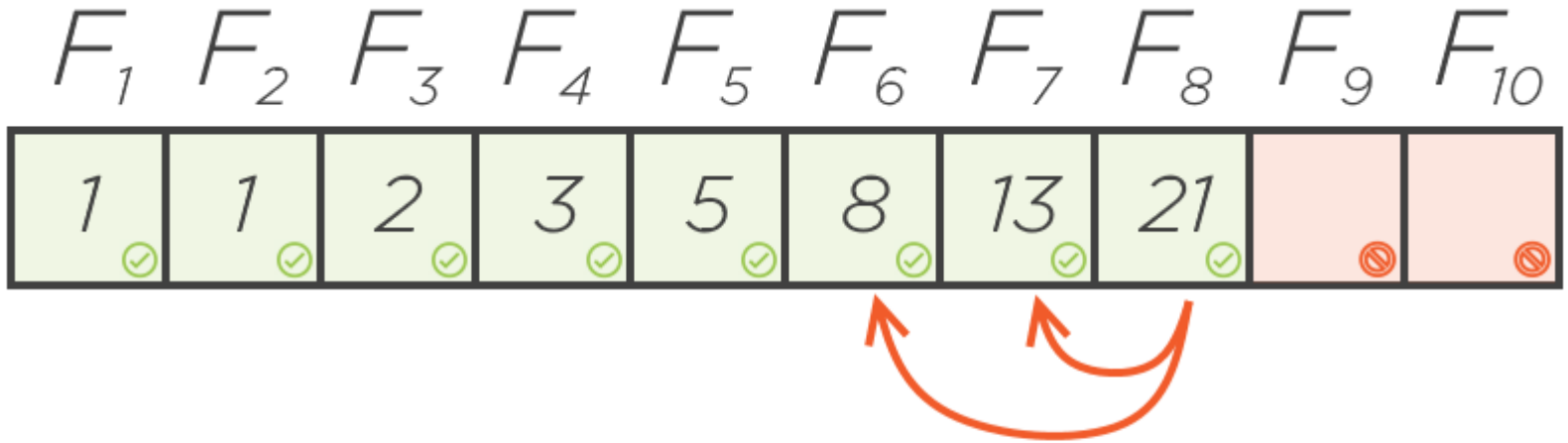
Dynamic Programming Defined



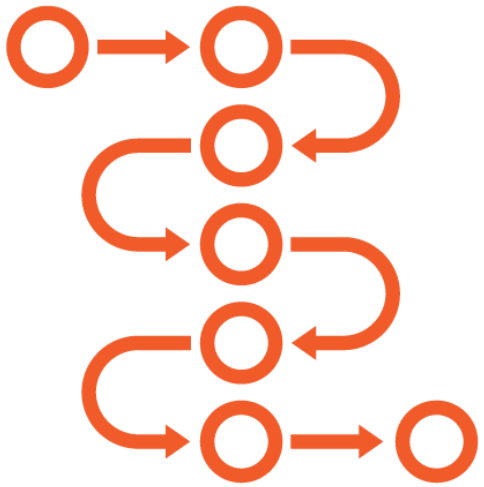


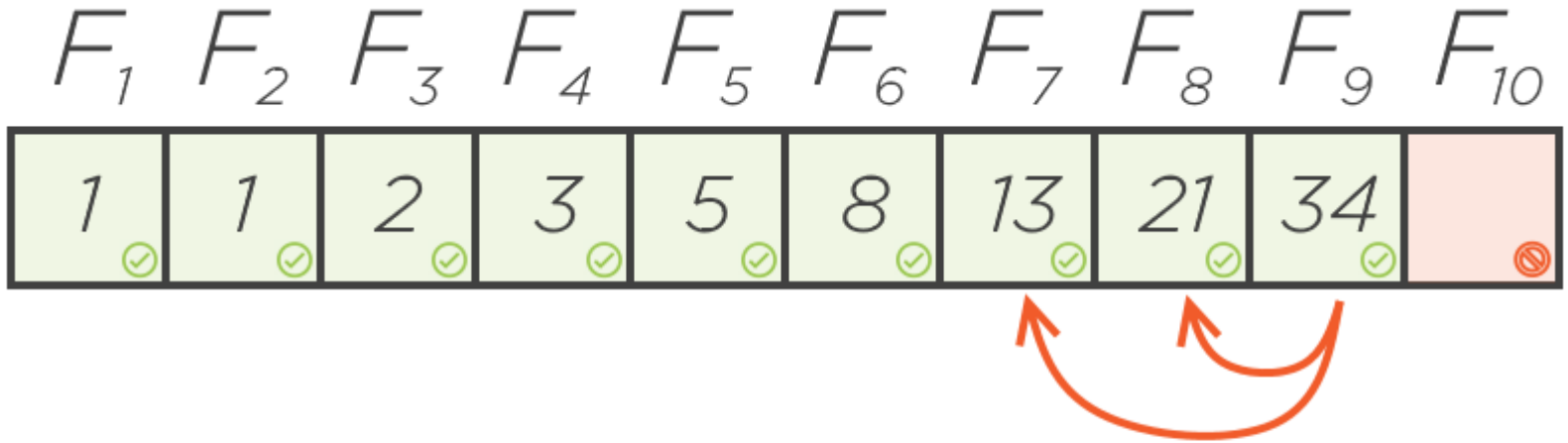
Dynamic Programming Defined



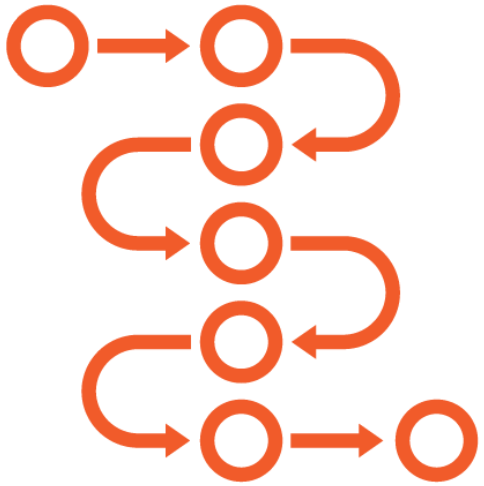


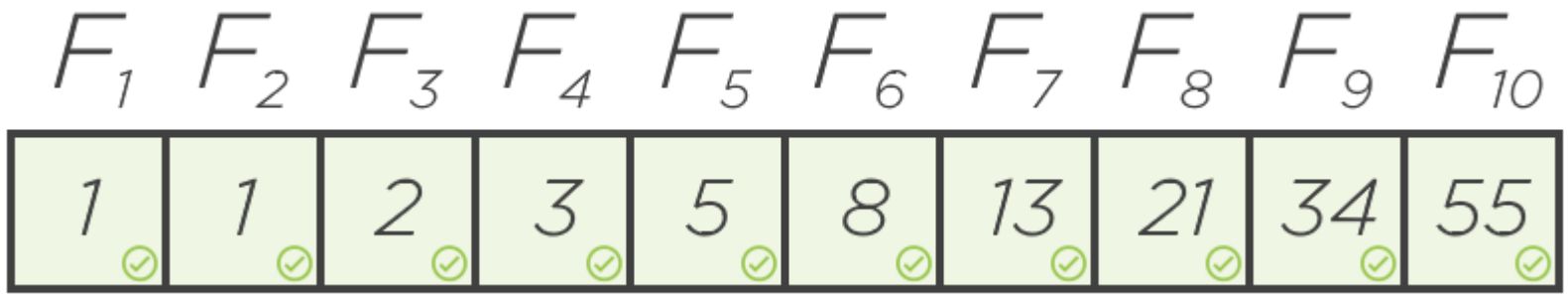
Dynamic Programming Defined





Dynamic Programming Defined

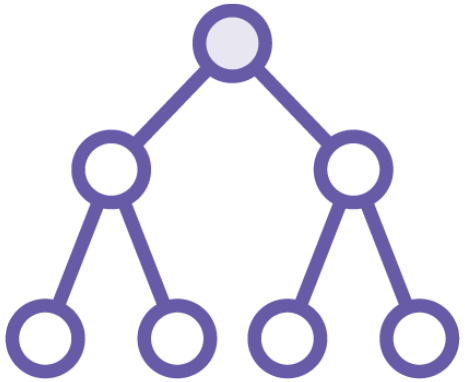




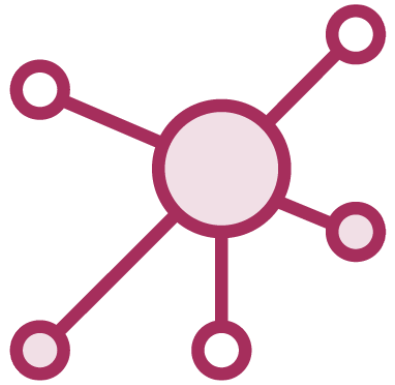
$F_{10}=55$

Memoization
as a special kind of
Dynamic Programming

Dynamic Programming

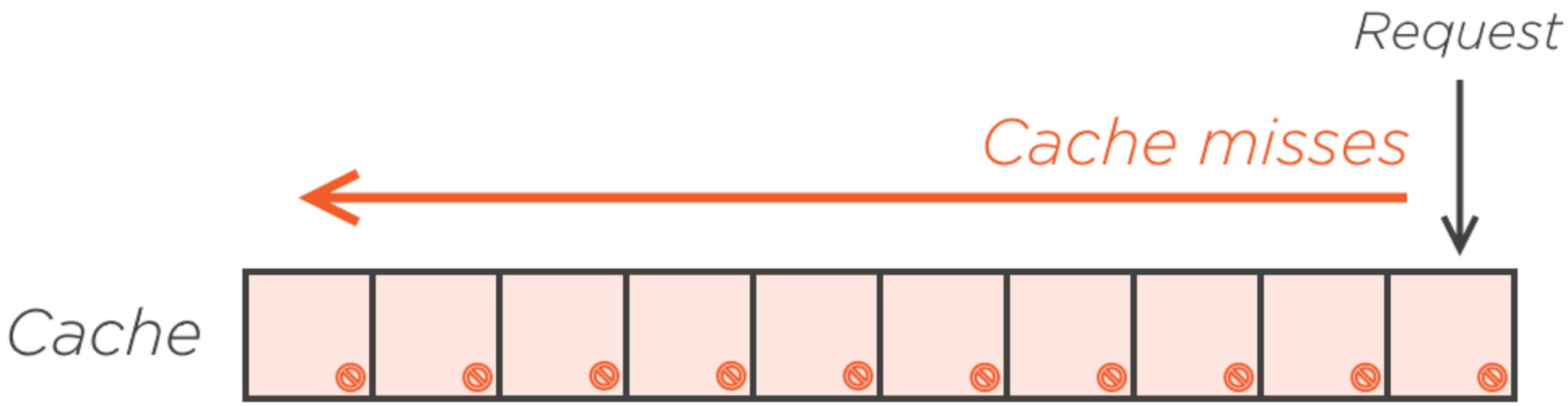


Memoization is
a special case of
Dynamic Programming

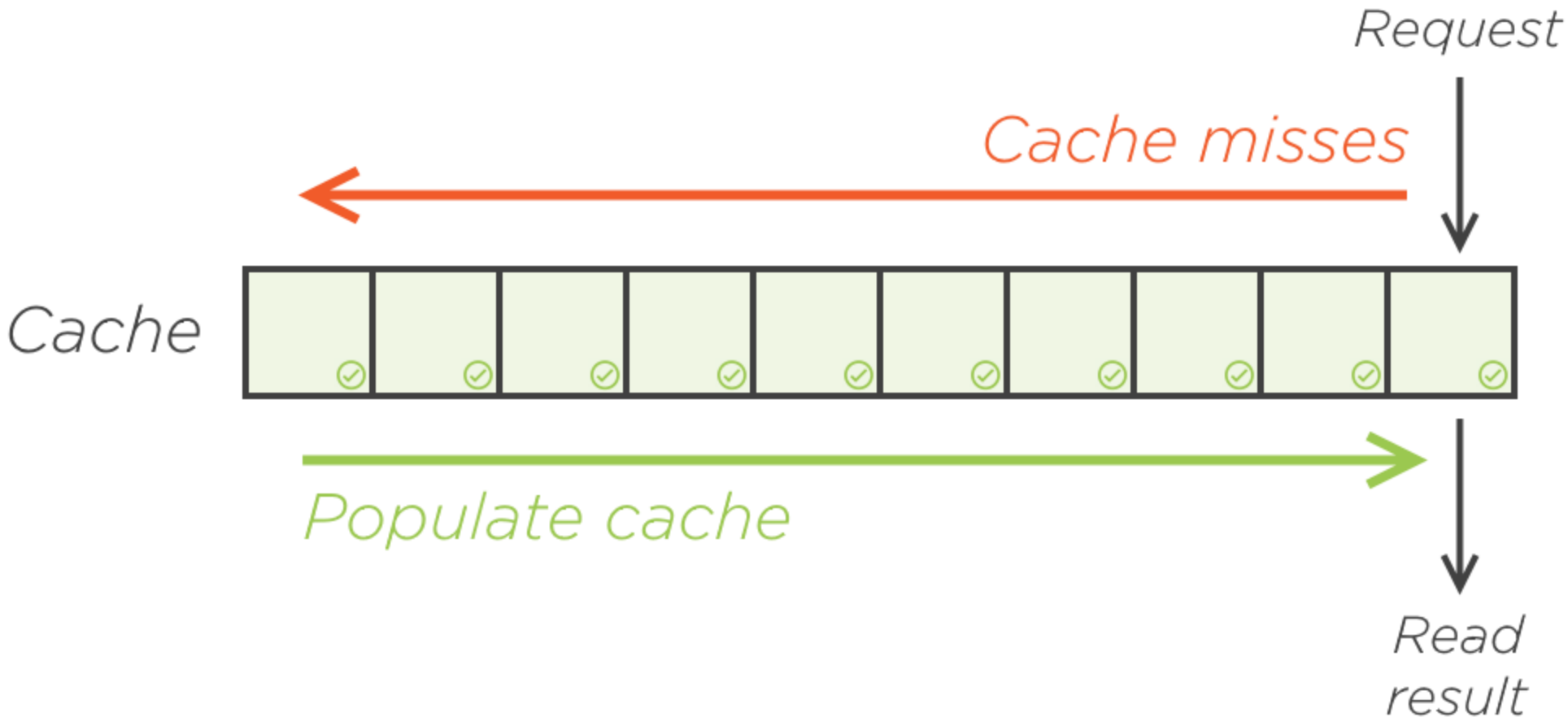


Dynamic Programming is
a generalization of
Memoization

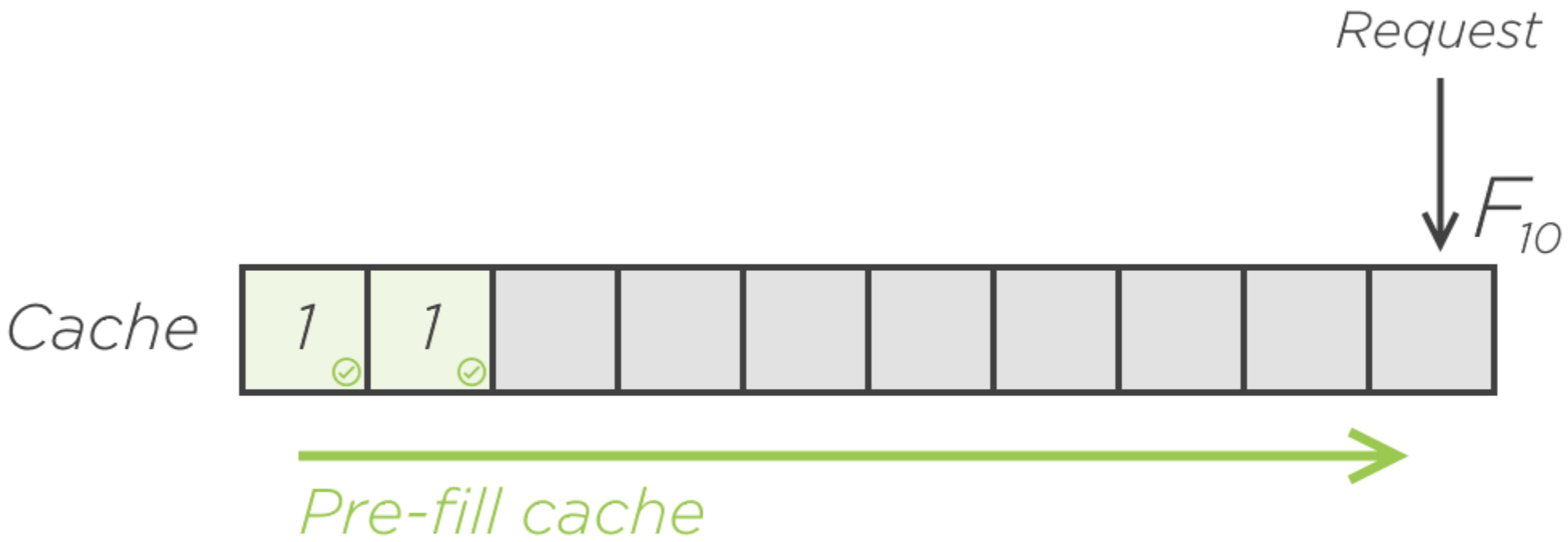
Dynamic Programming



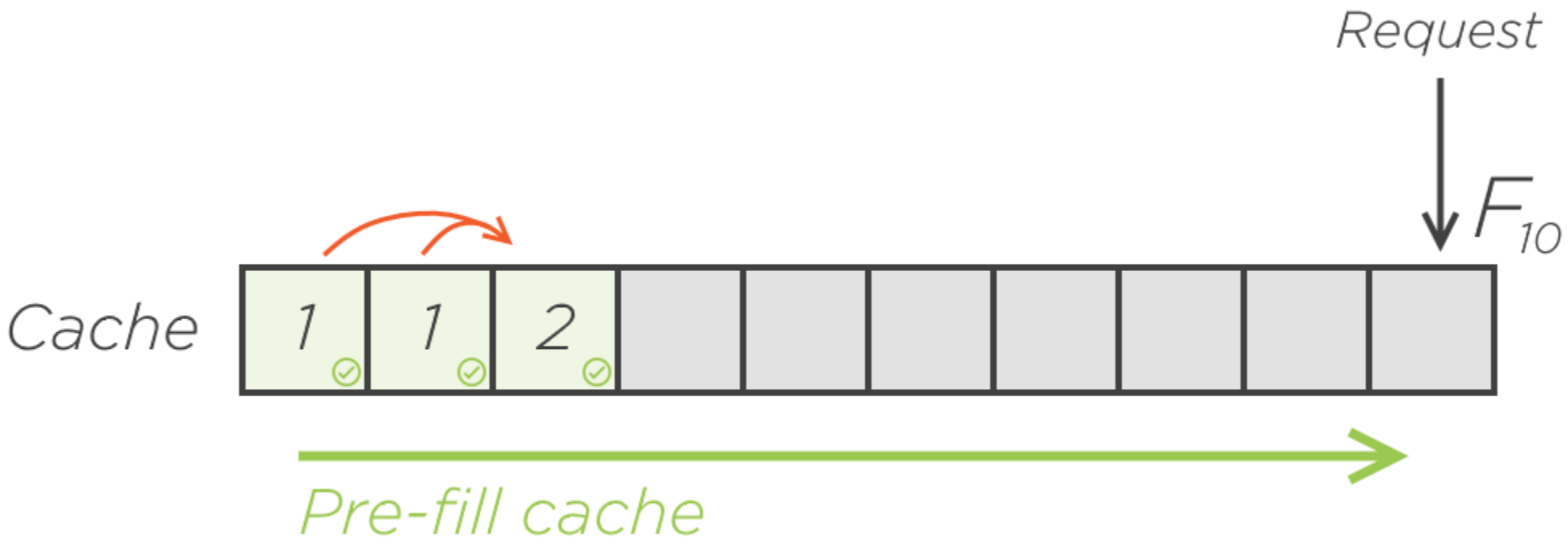
Dynamic Programming



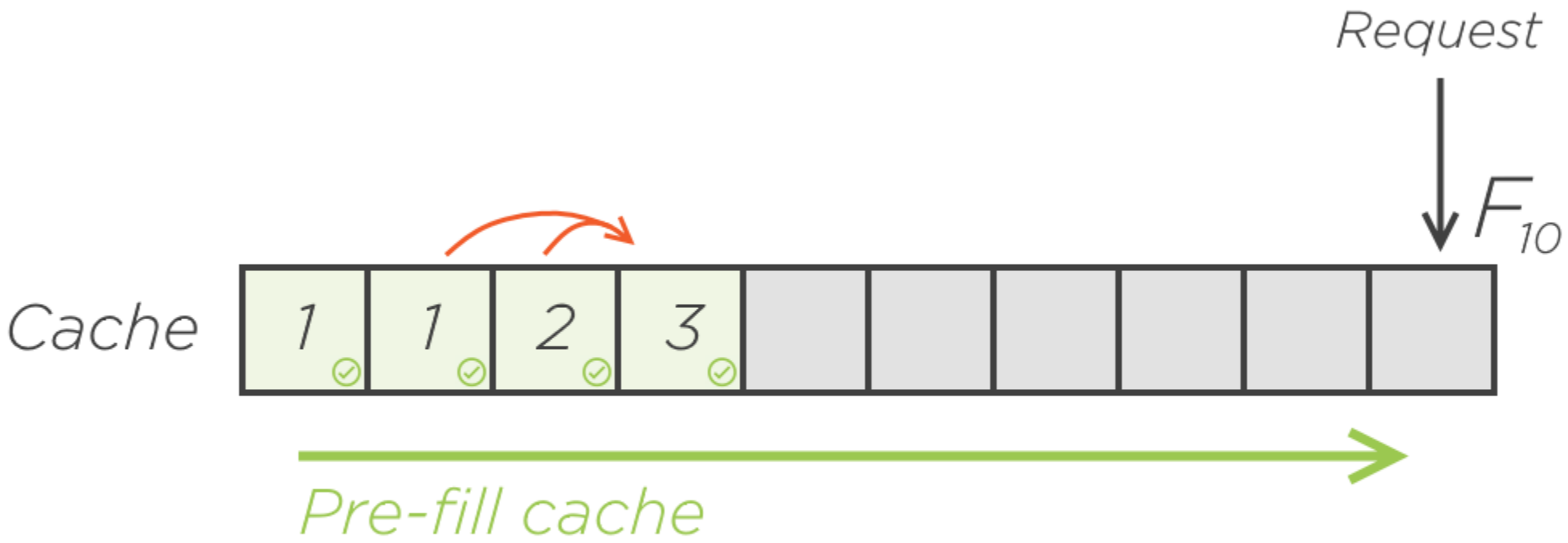
Dynamic Programming



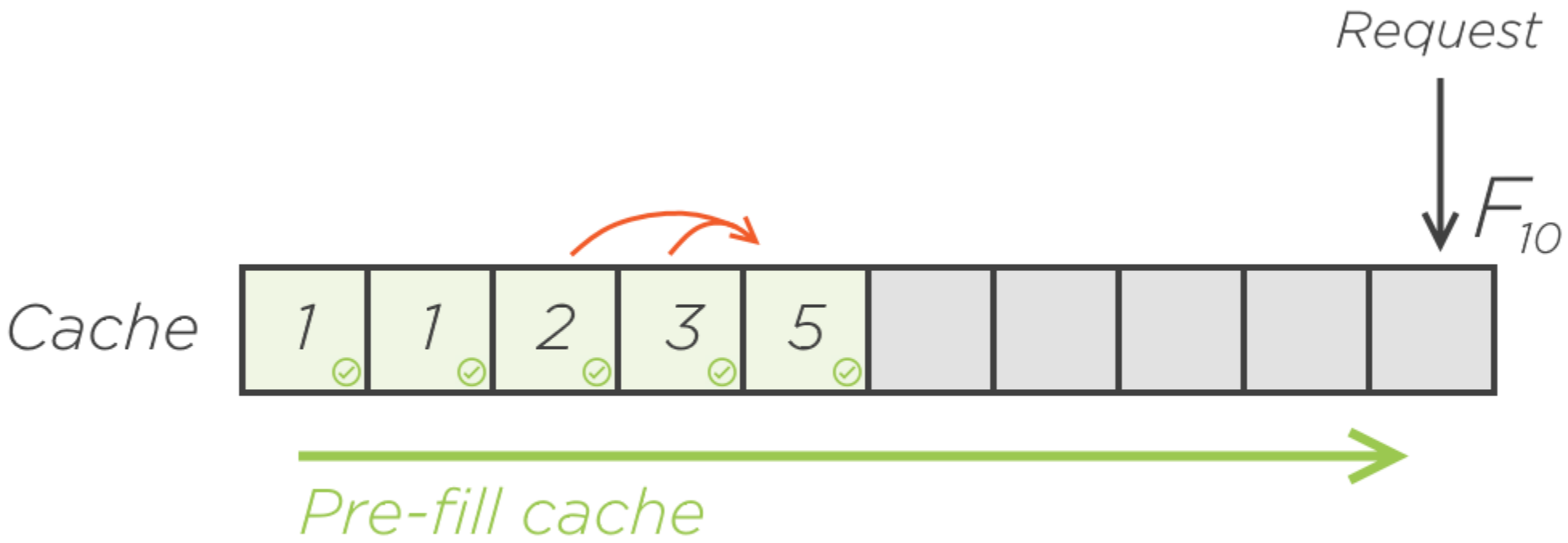
Dynamic Programming



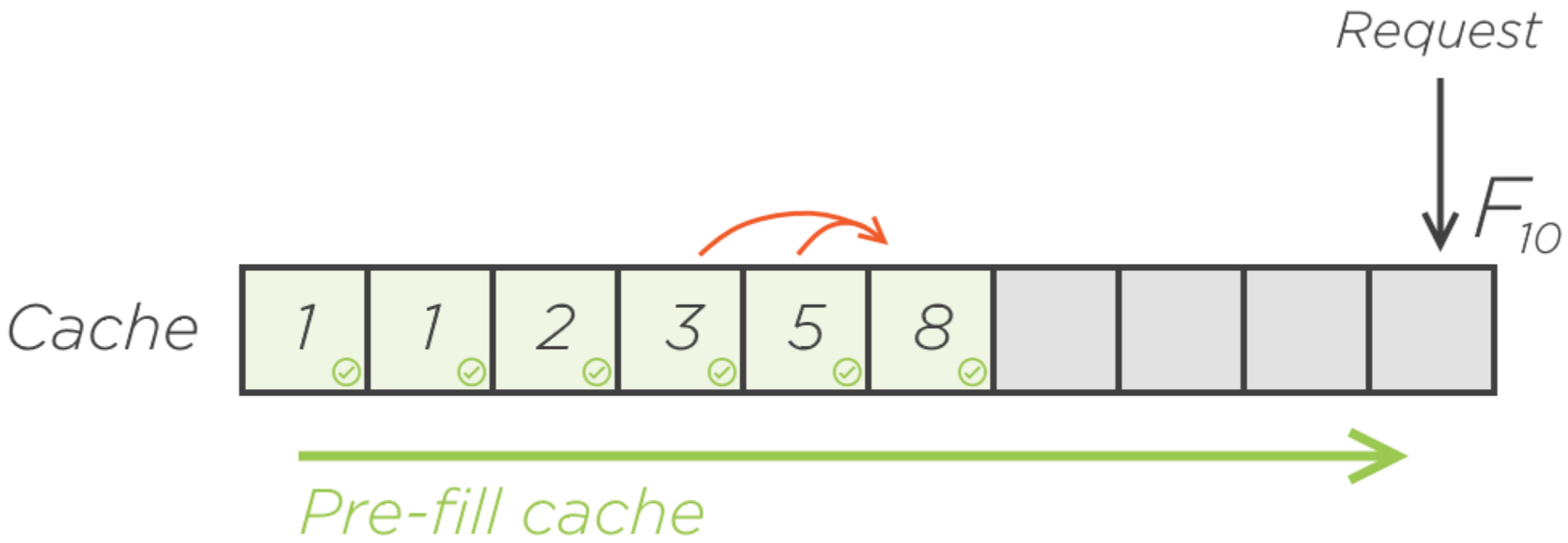
Dynamic Programming



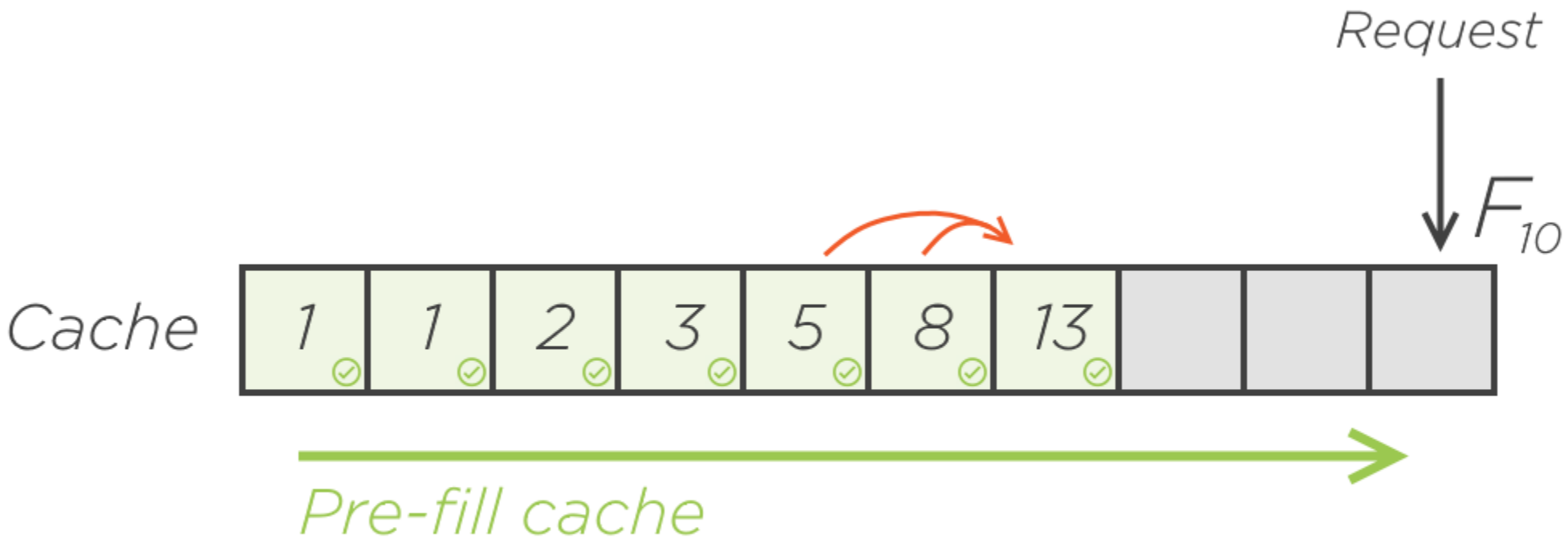
Dynamic Programming



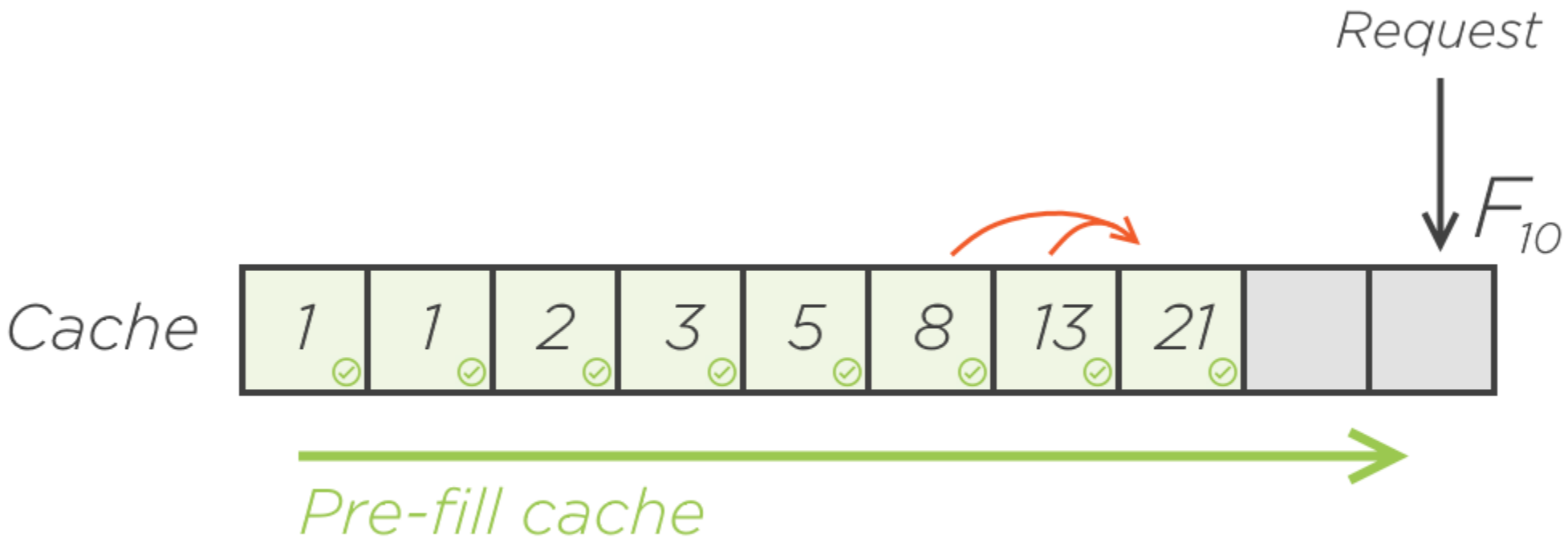
Dynamic Programming



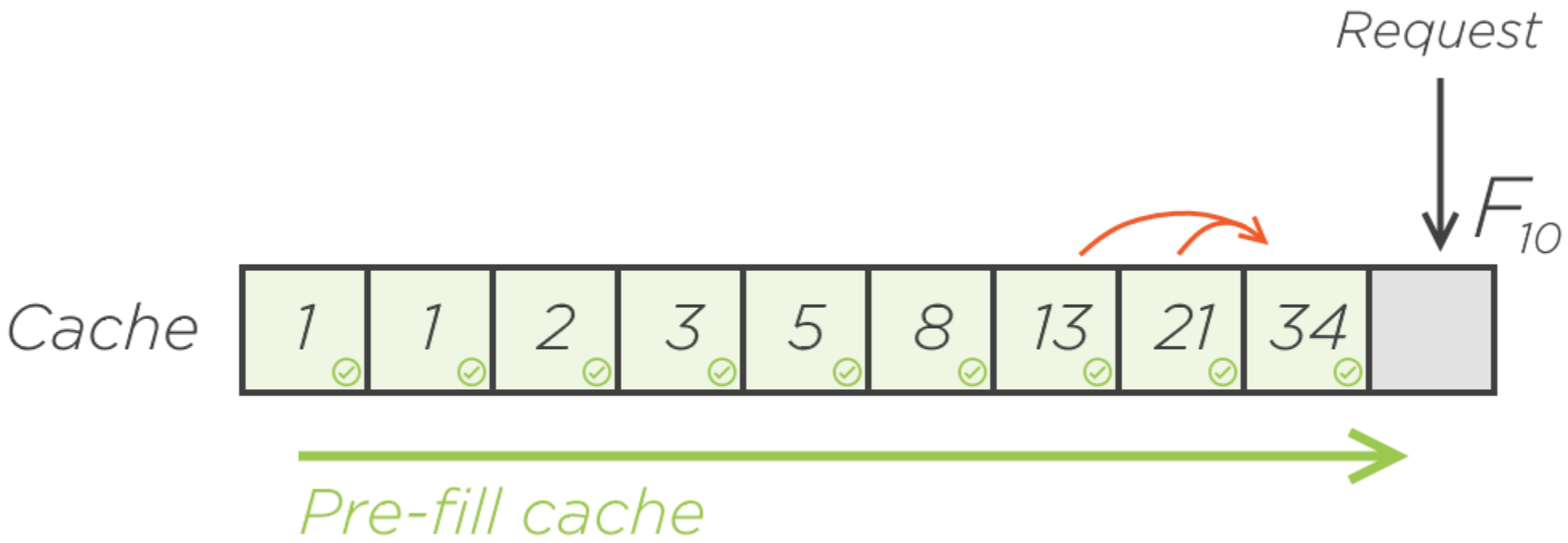
Dynamic Programming



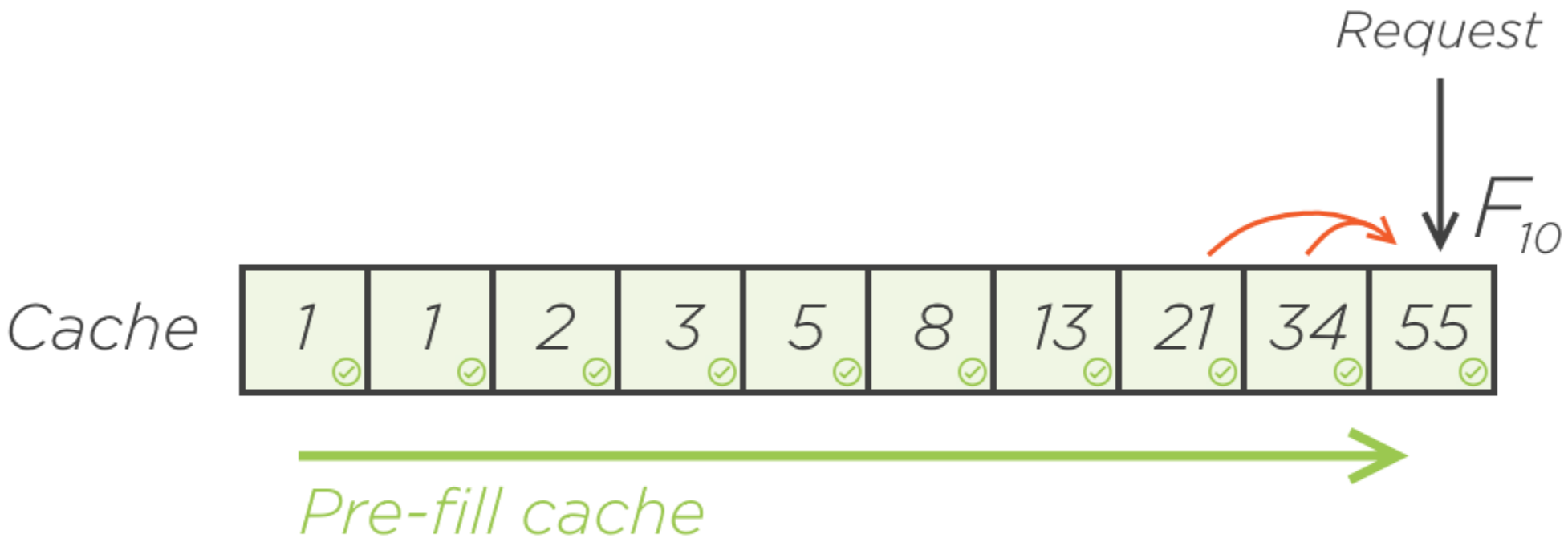
Dynamic Programming



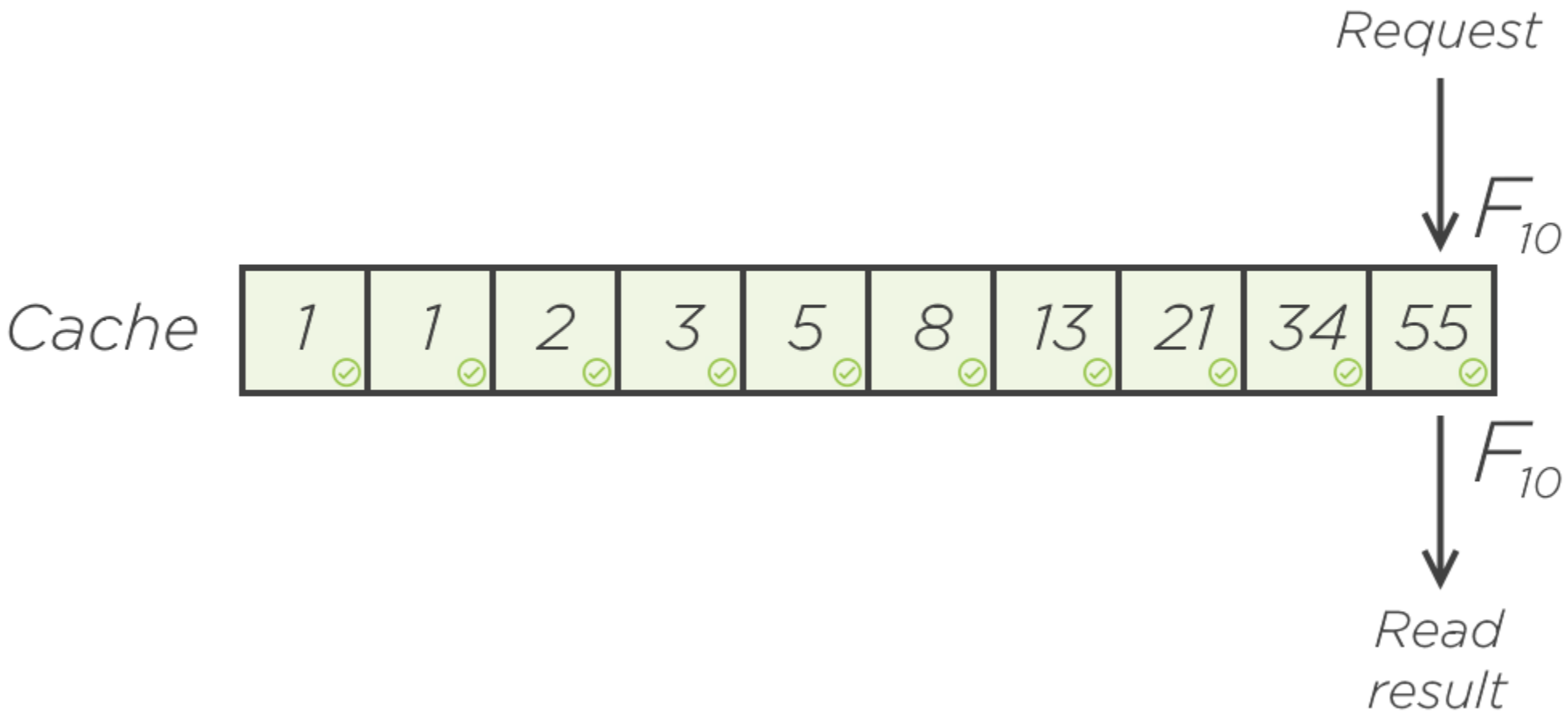
Dynamic Programming



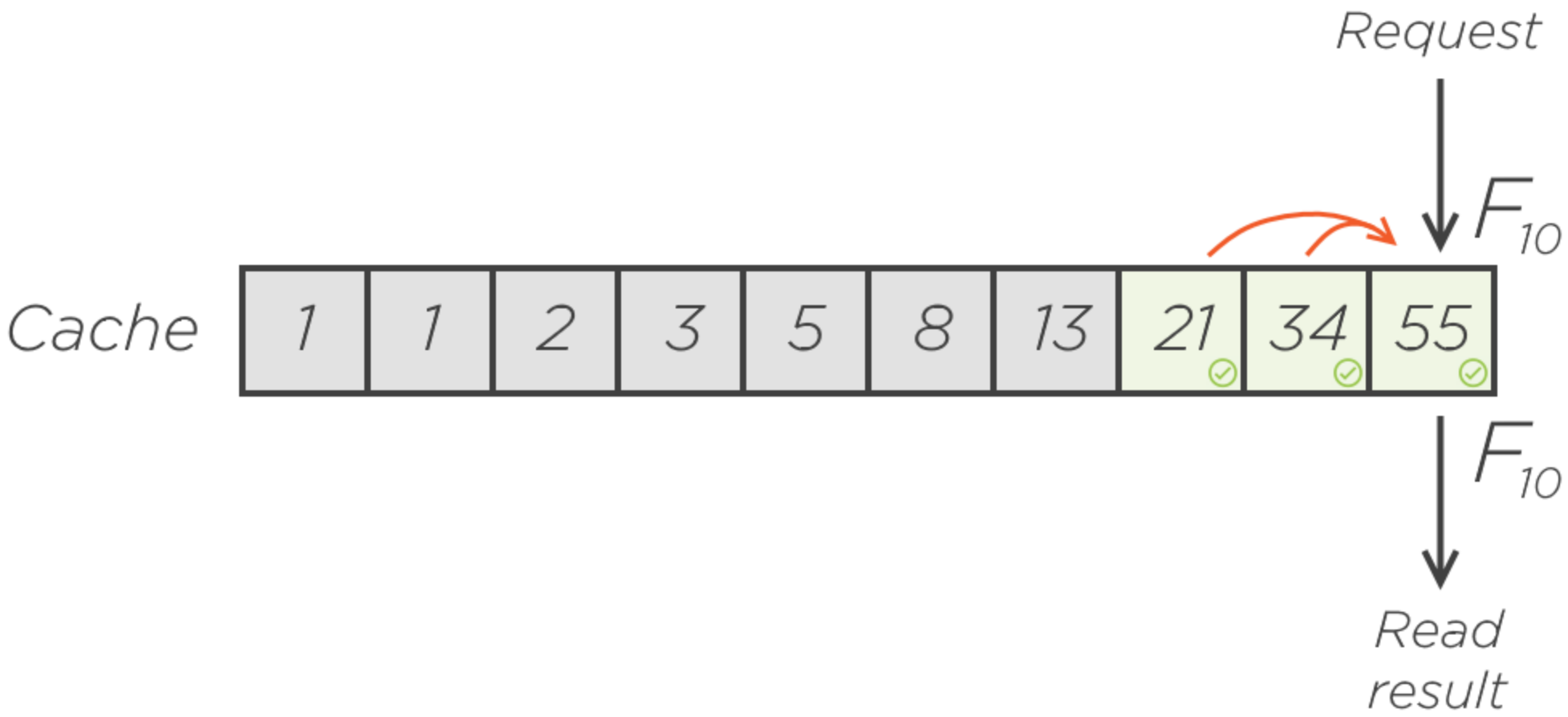
Dynamic Programming



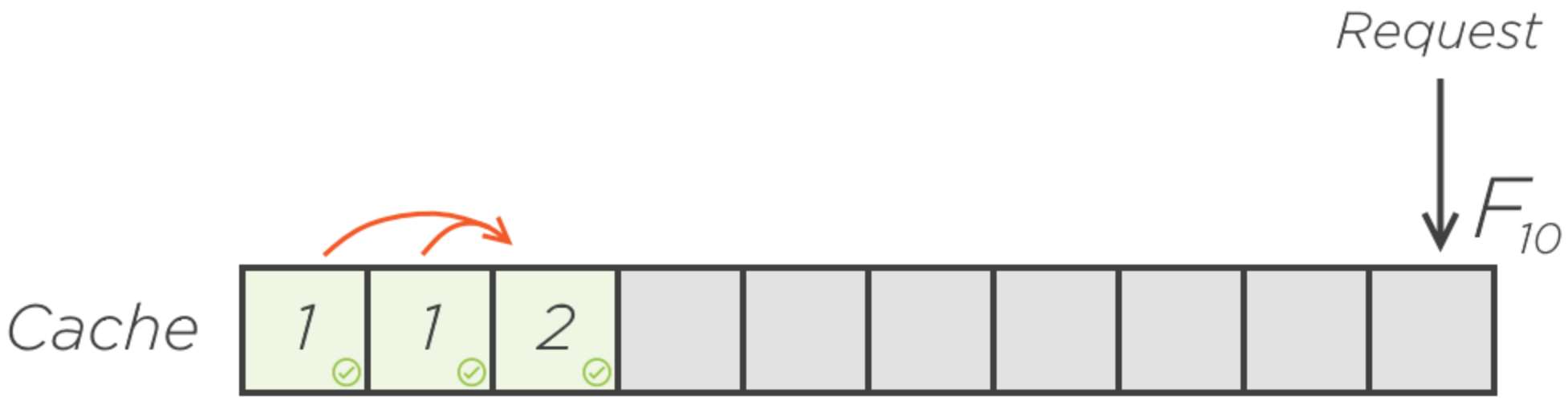
Dynamic Programming



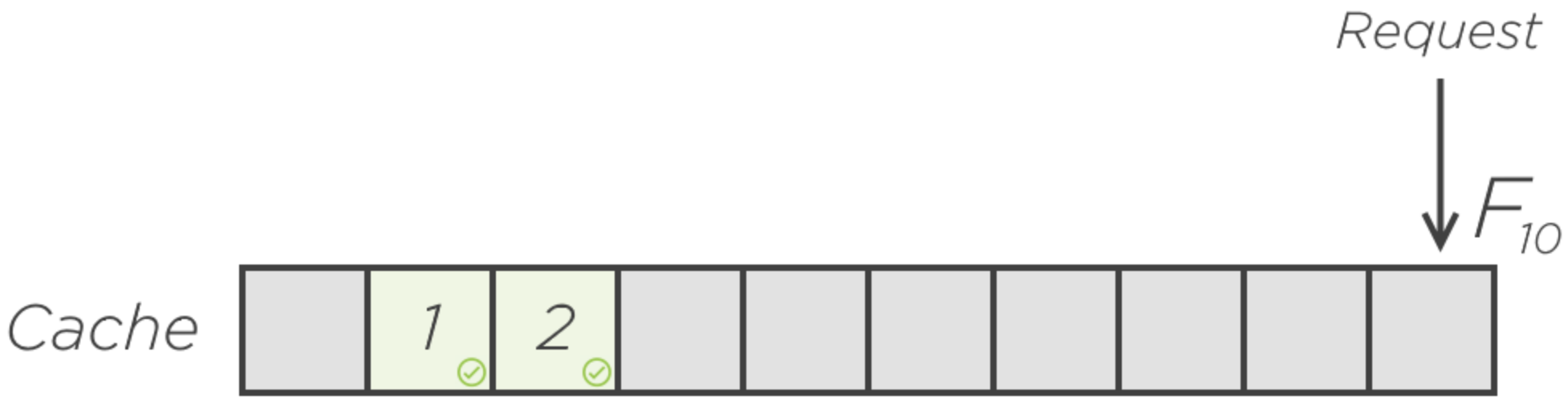
Inventing an Algorithm



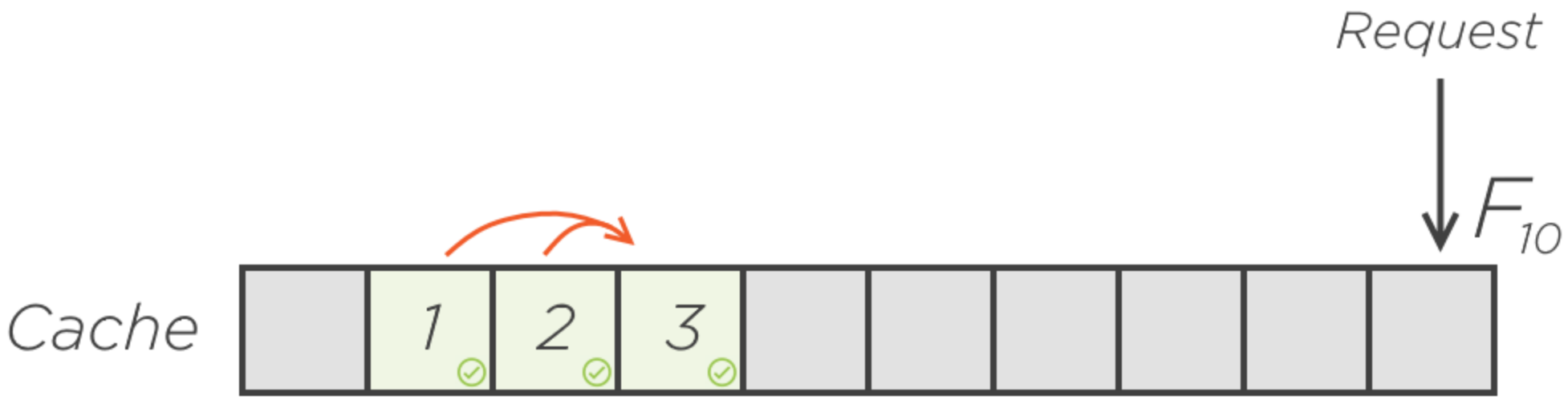
Inventing an Algorithm



Inventing an Algorithm

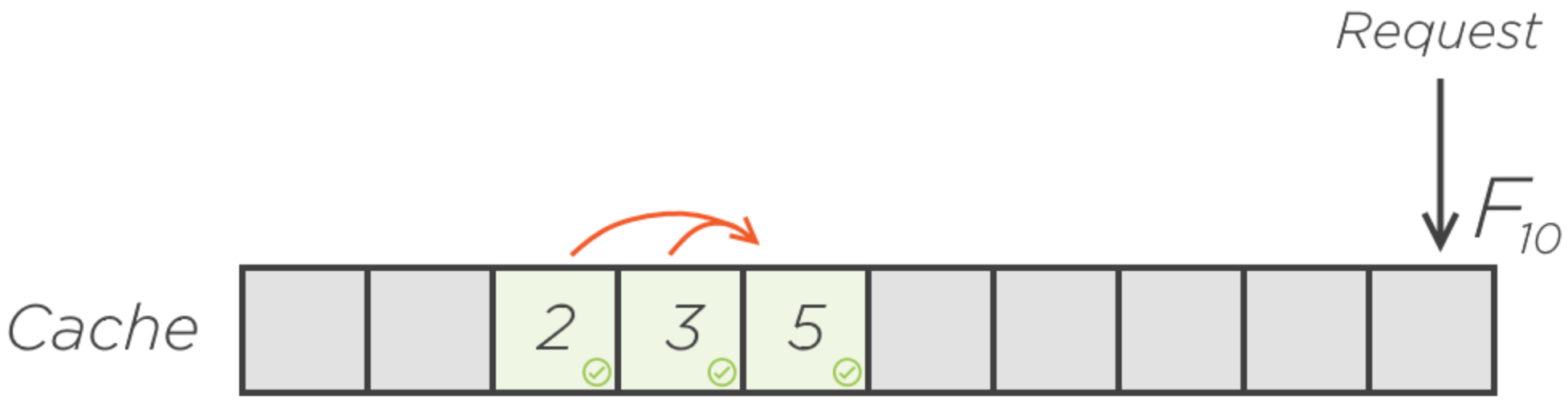


Inventing an Algorithm



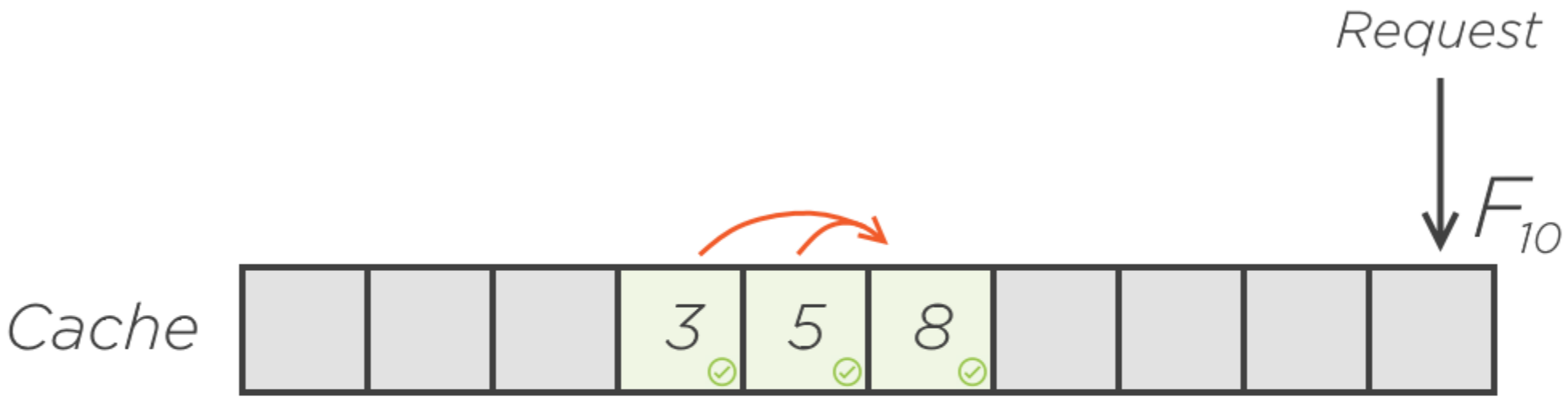
Inventing an Algorithm

Inventing an Algorithm

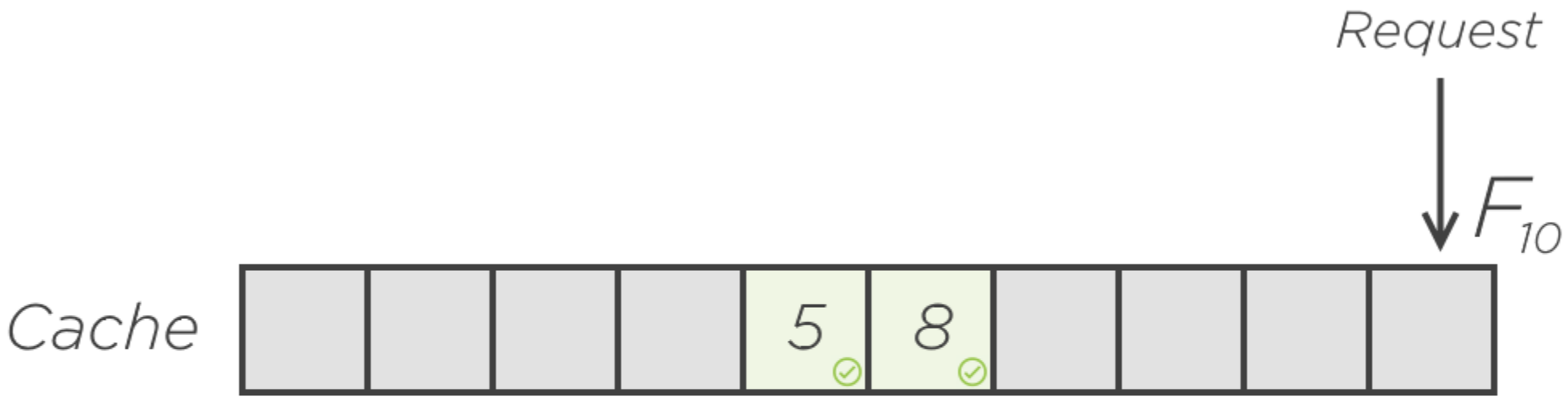


Inventing an Algorithm

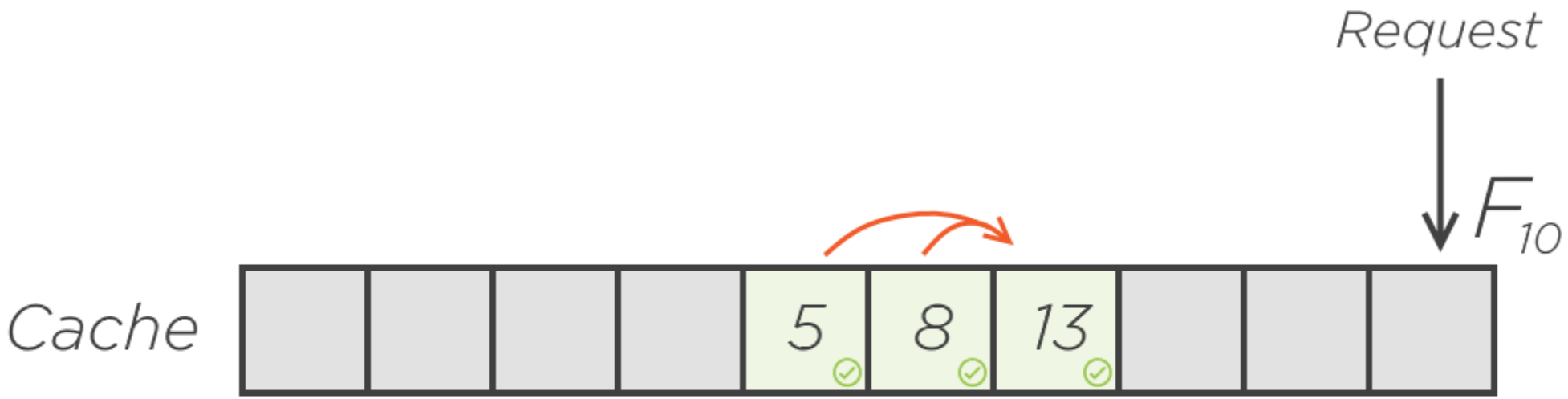
Inventing an Algorithm



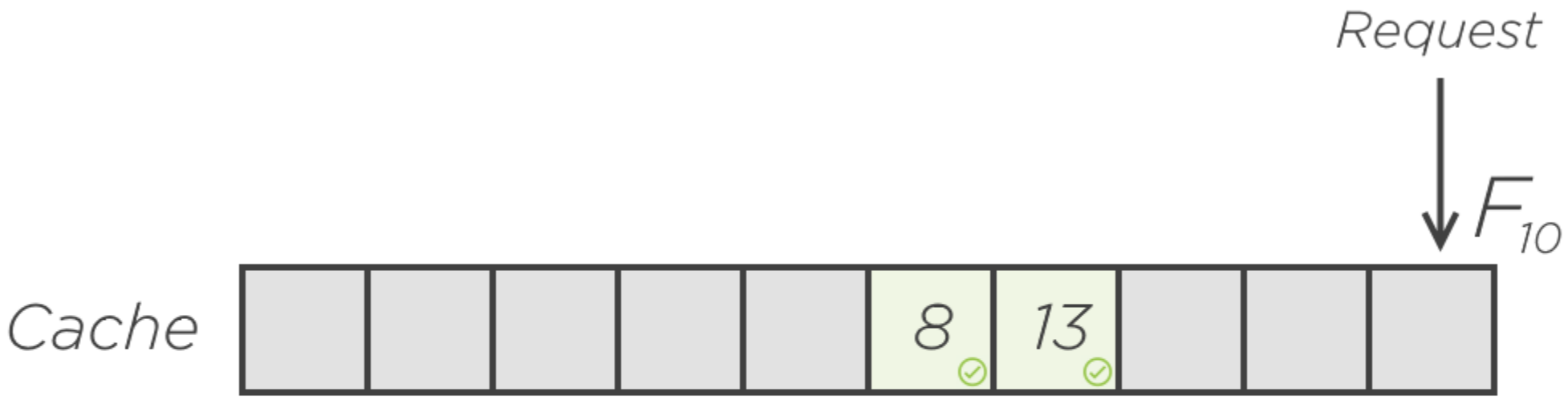
Inventing an Algorithm



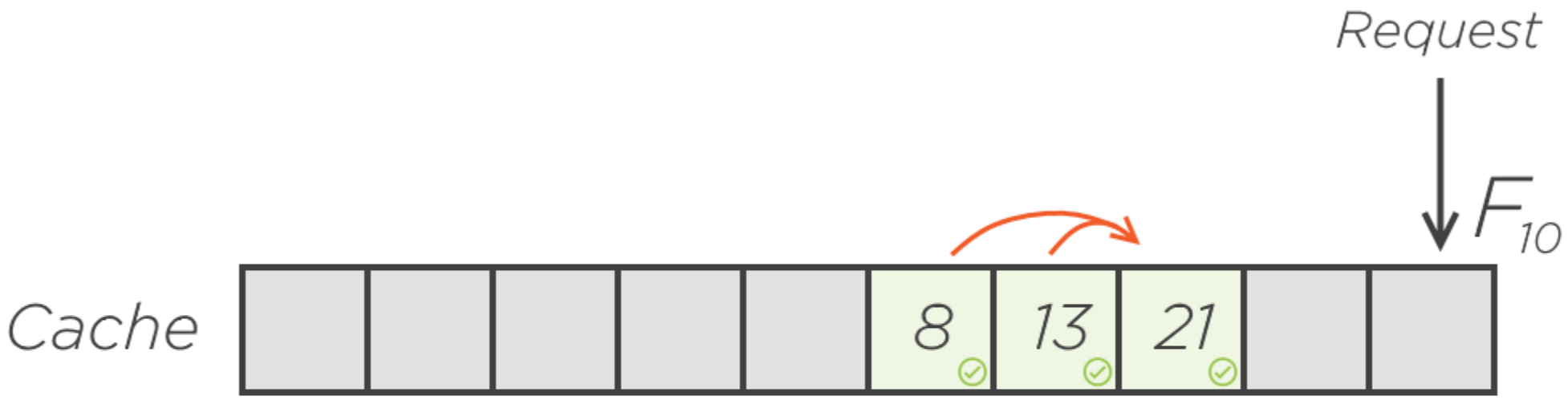
Inventing an Algorithm



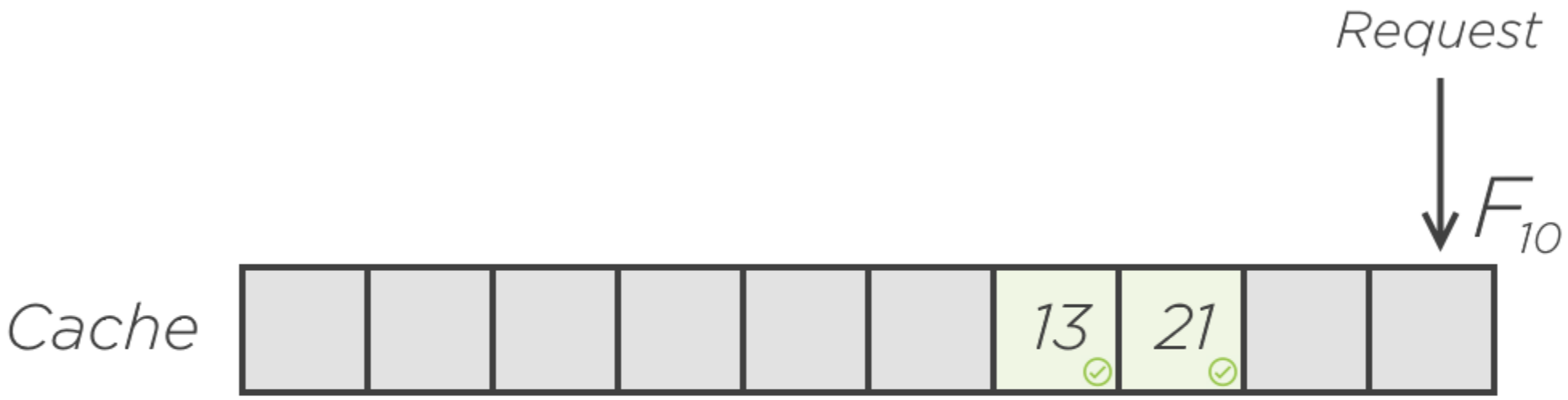
Inventing an Algorithm



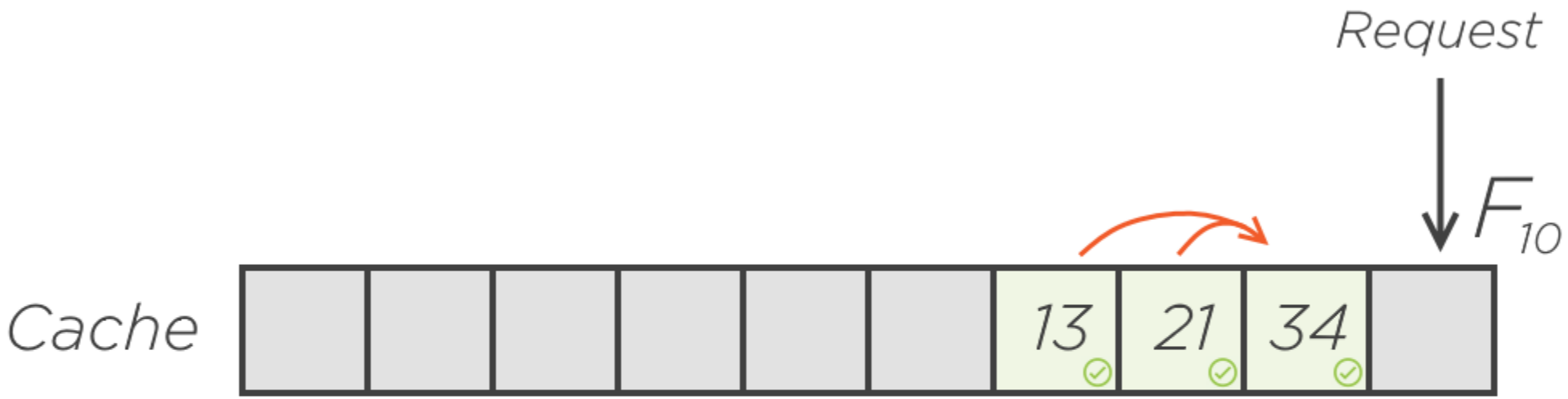
Inventing an Algorithm



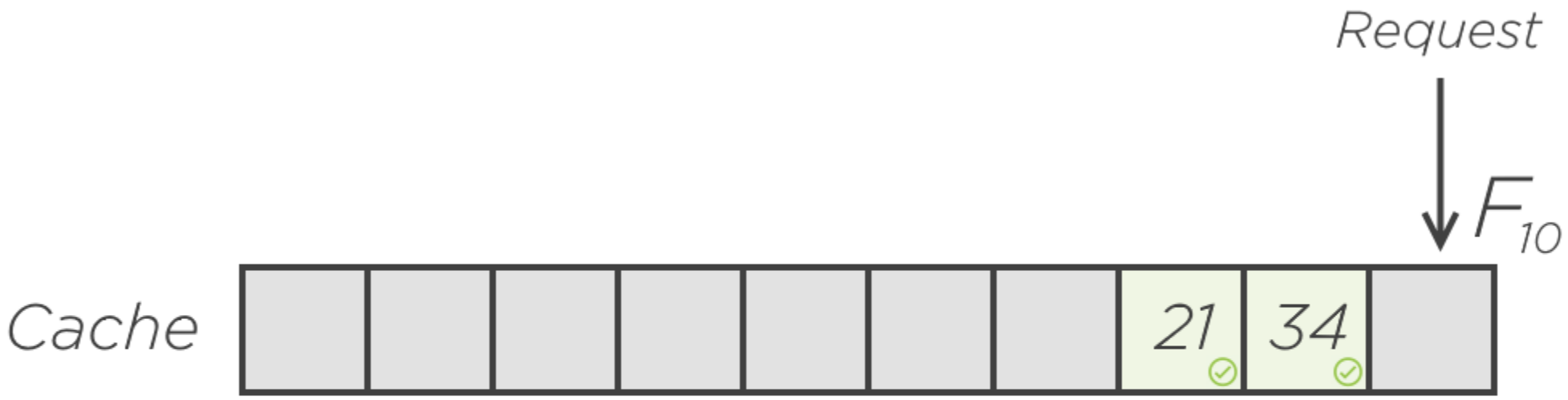
Inventing an Algorithm



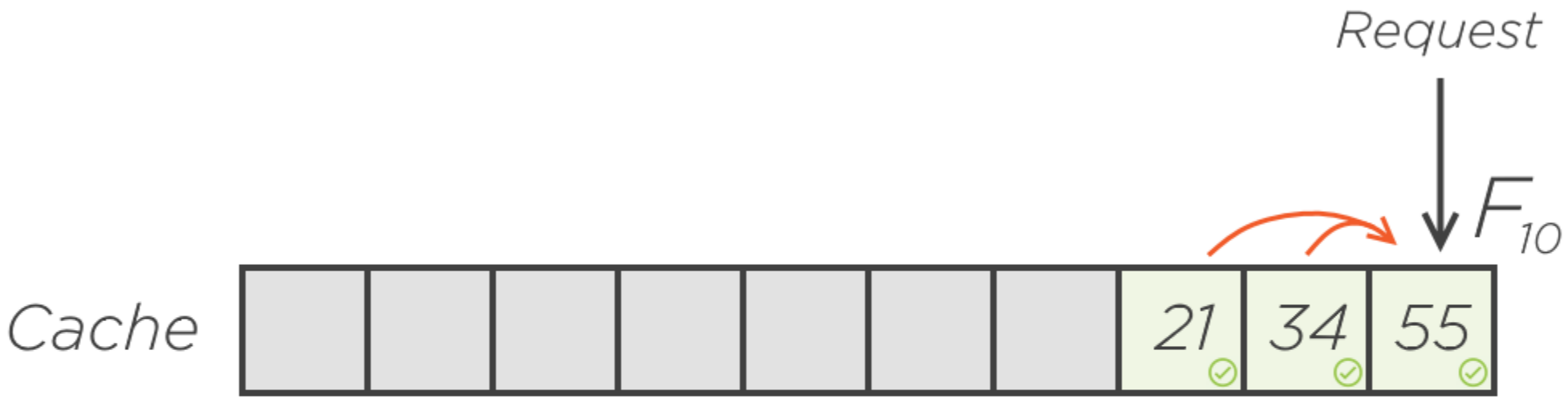
Inventing an Algorithm



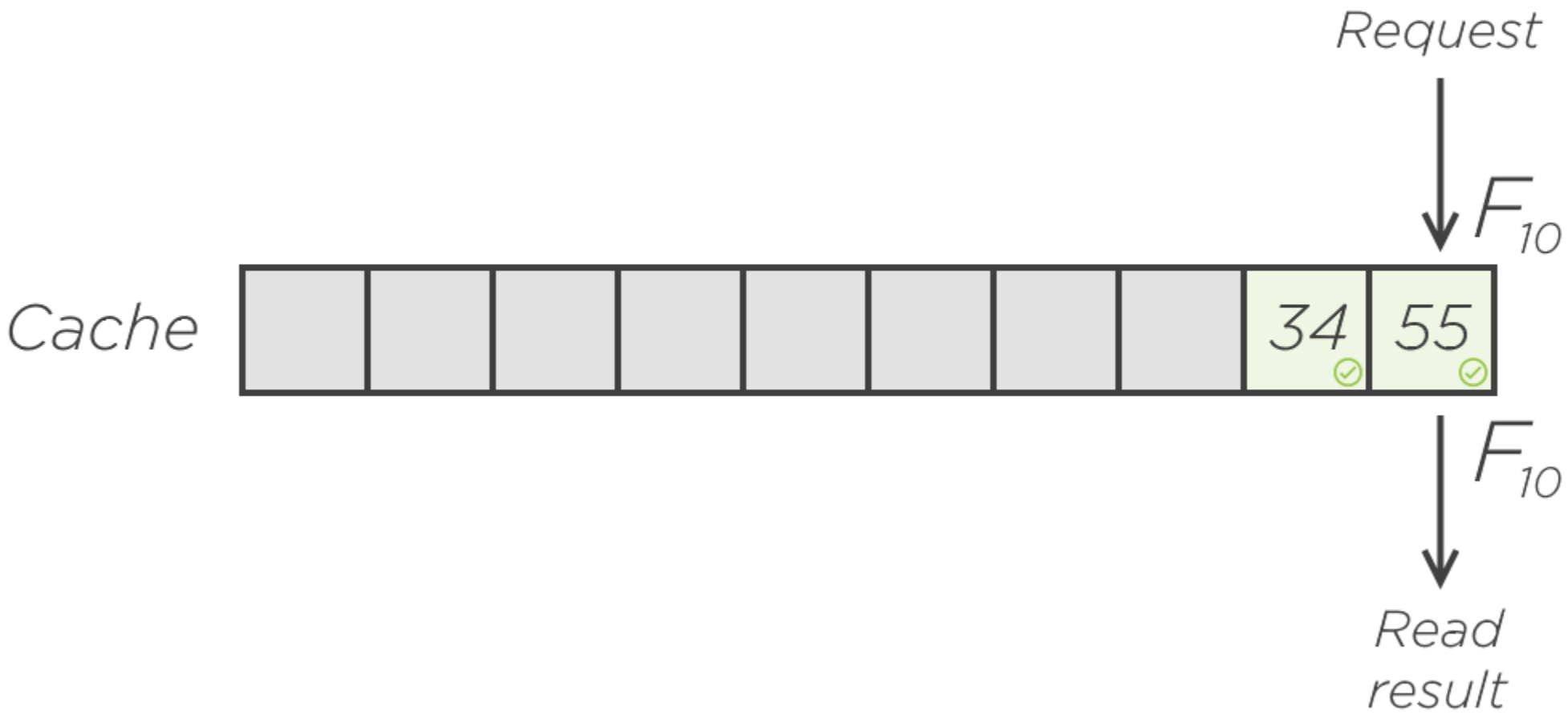
Inventing an Algorithm



Inventing an Algorithm



Inventing an Algorithm



Leveraging Memoization

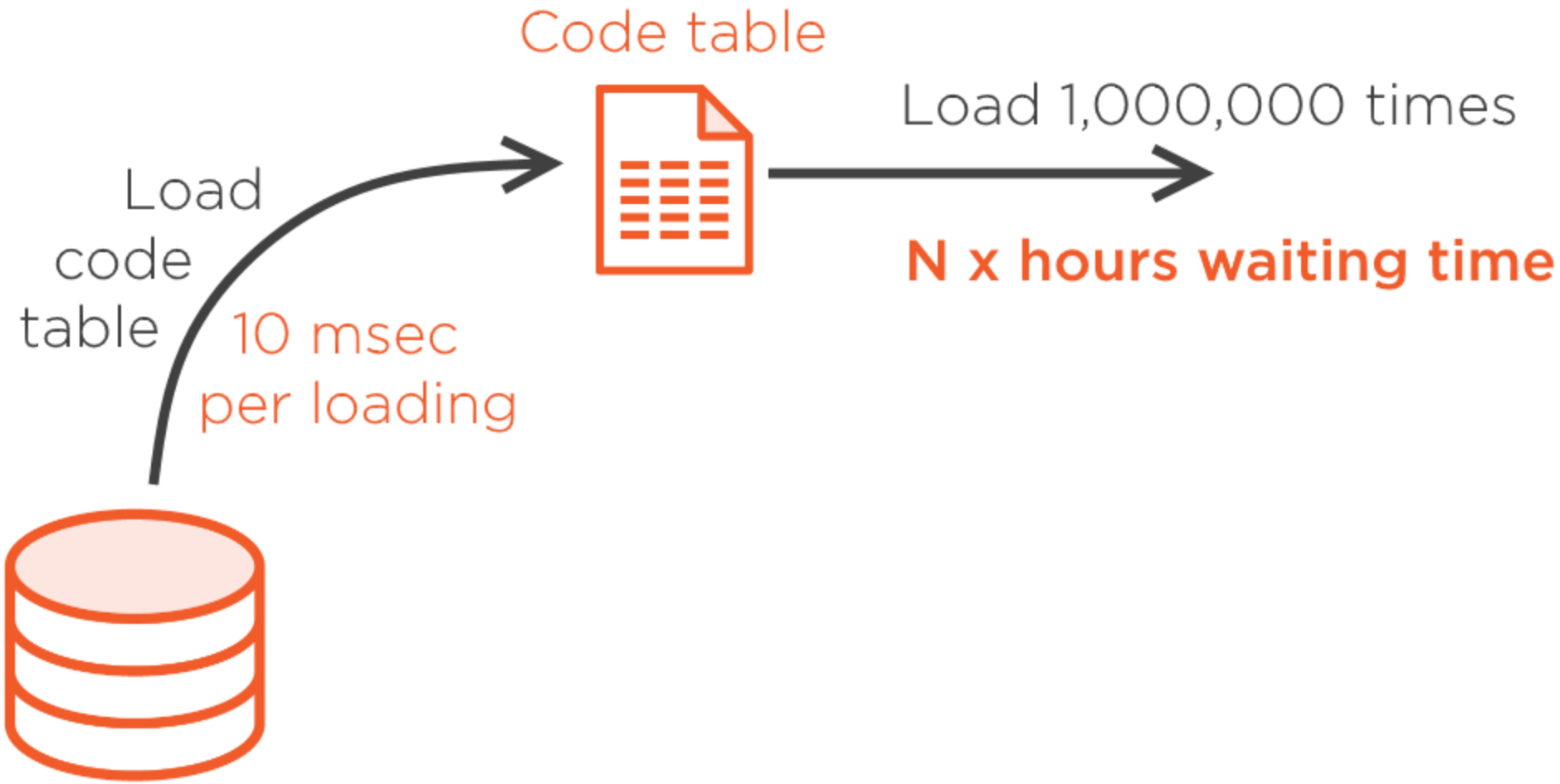
Only applies to pure functions

Makes sense when there is noticeable performance drop

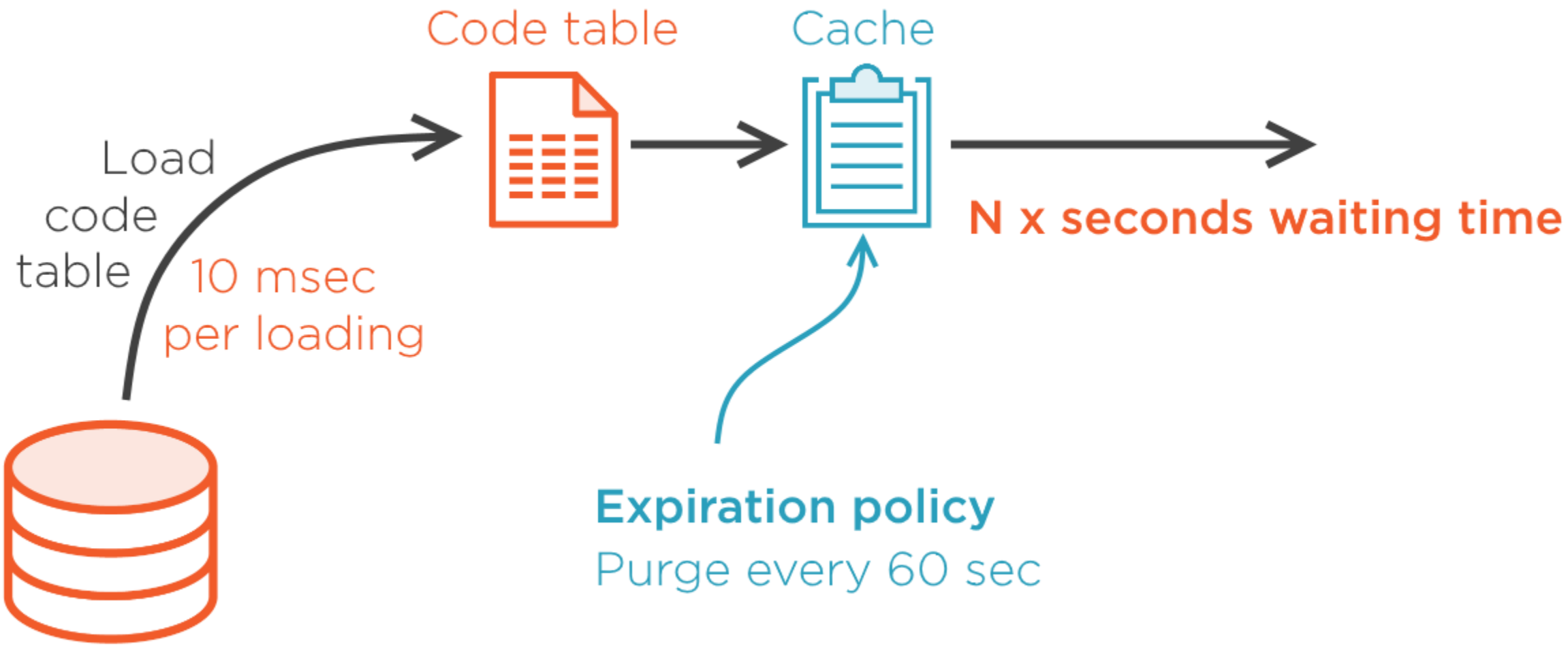
Trades time (CPU, I/O, etc.) for memory

Don’t forget to measure effects

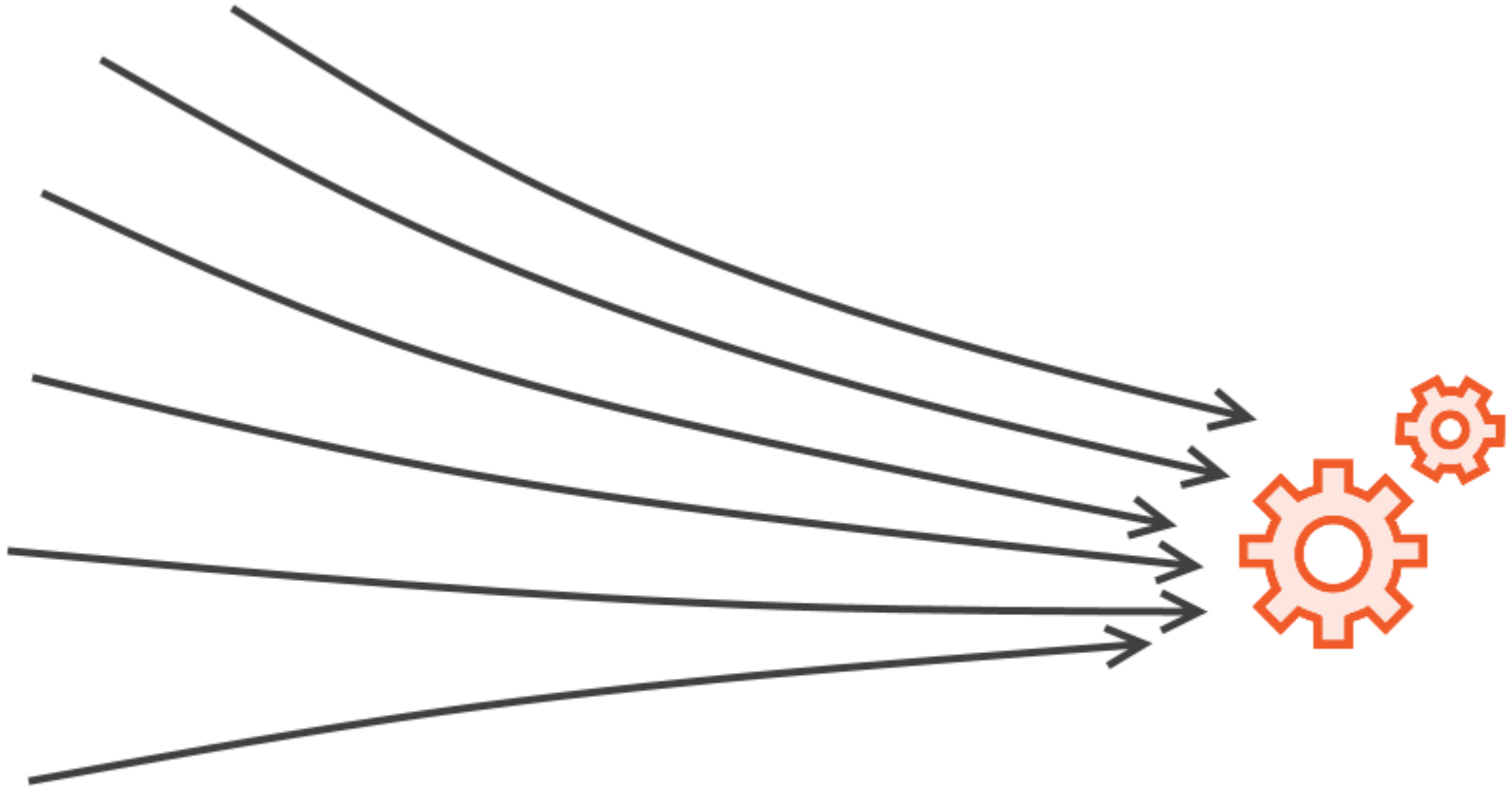
Leveraging Memoization



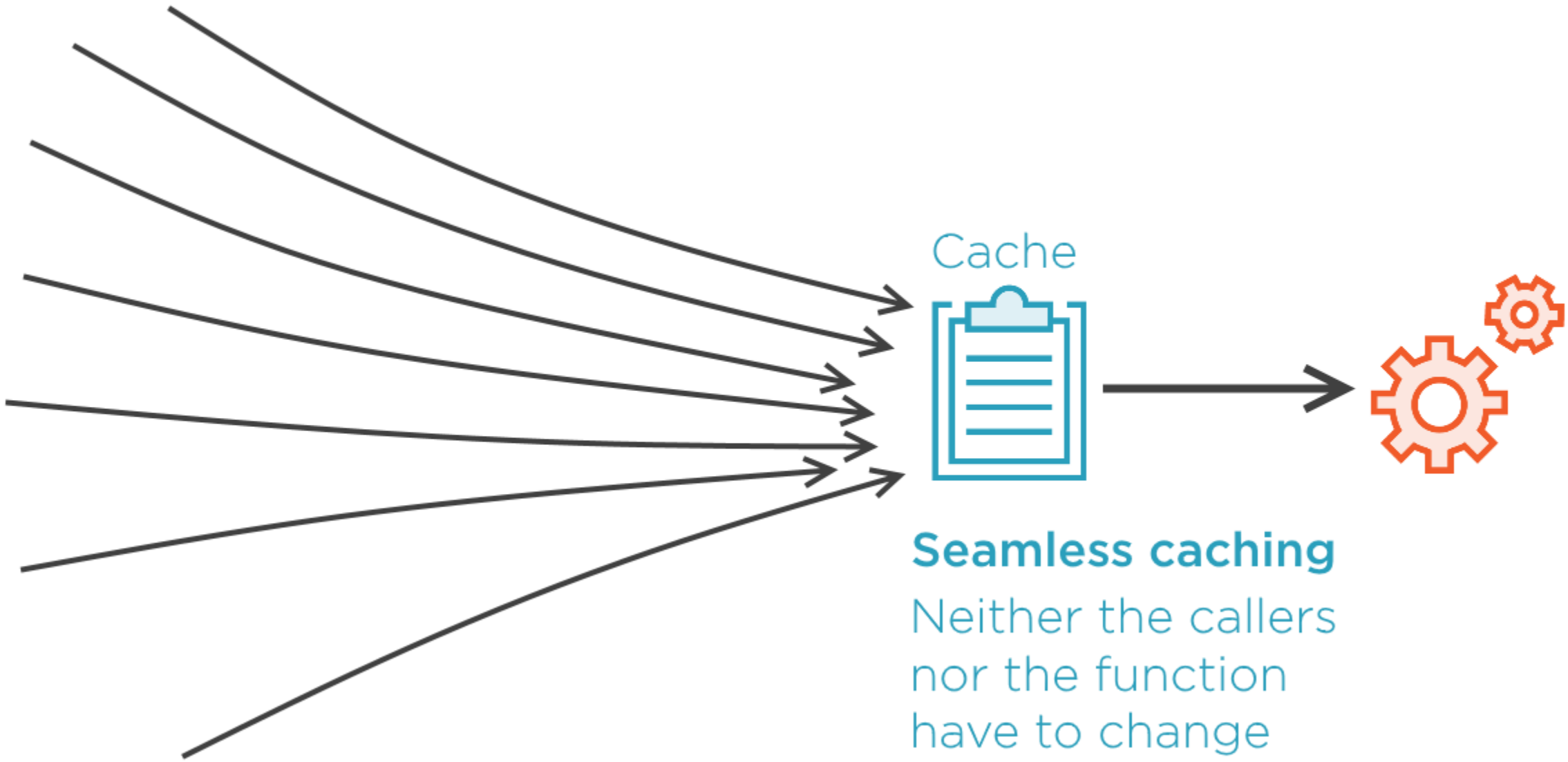
Leveraging Memoization



Leveraging Memoization



Leveraging Memoization



Seamless caching

Neither the callers nor the function have to change

Summary

Memoization of pure functions

- Special case of Dynamic Programming
- Every subproblem evaluated once
- Improves run time performance

General memoization

- Wrap a function and cache its results
- PostSharp defines caching attribute
- Becomes orthogonal concern

Next module:
Working with Pure Member Functions