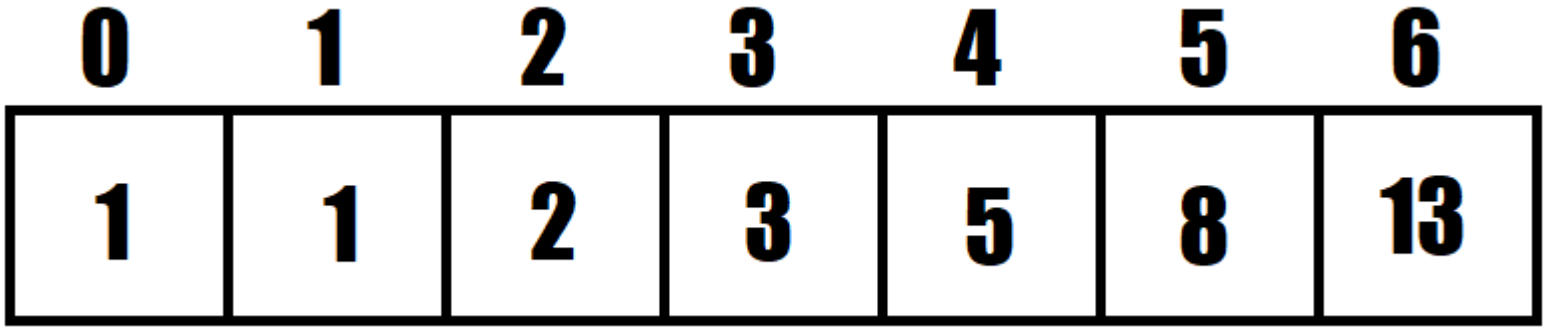
Programação dinâmica

Programação dinâmica

Aproveitar casos previamente calculados para resolver um problema maior.

Sequência de Fibonacci

Fib[i]: representa o i-ésimo termo da sequencia de Fibonacci



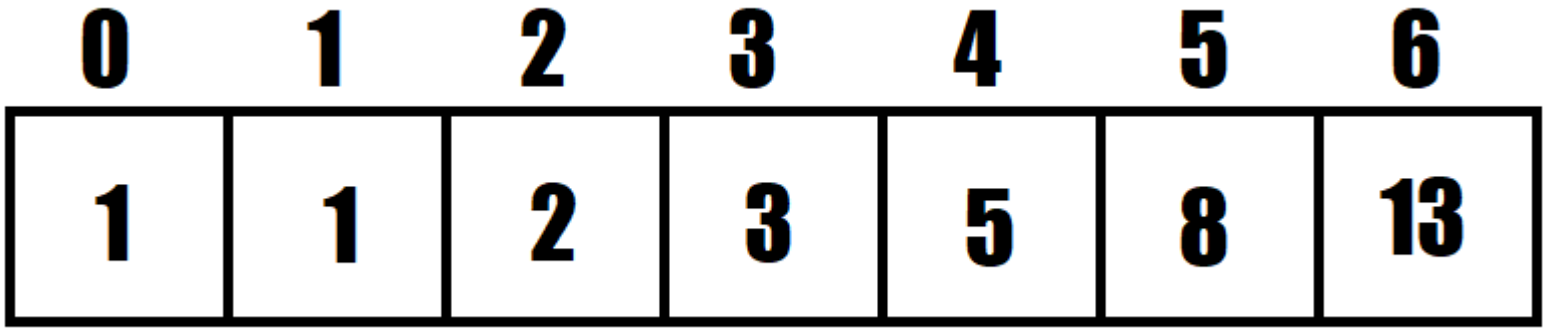
Sequência de Fibonacci

Casos base:

Fib[0] = Fib[1] = 1;

Caso geral:

Fib[i] = Fib[i-1] + Fib[i-2];



Sequência de Fibonacci

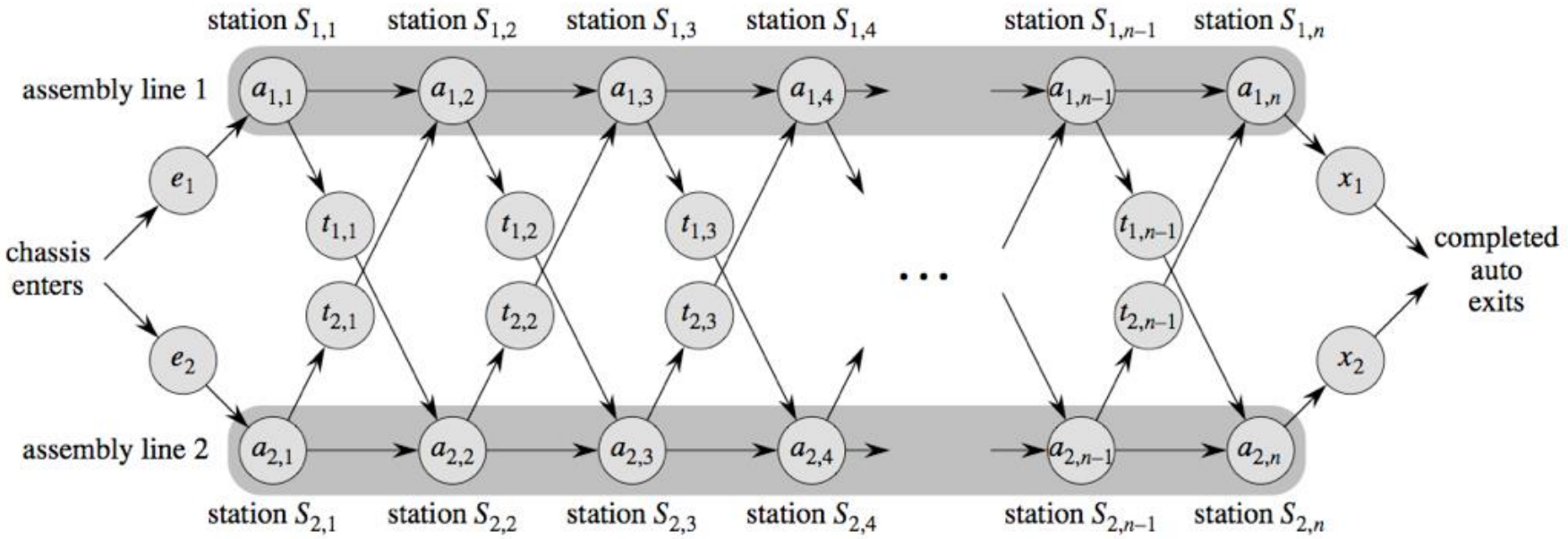
```
int Fib[100];
Fib[0] = 0;
Fib[1] = 1;

for(int i = 2; i <= n; i++)
    Fib[i] = Fib[i-1] + Fib[i-2];

cout << Fib[n] << endl;
```

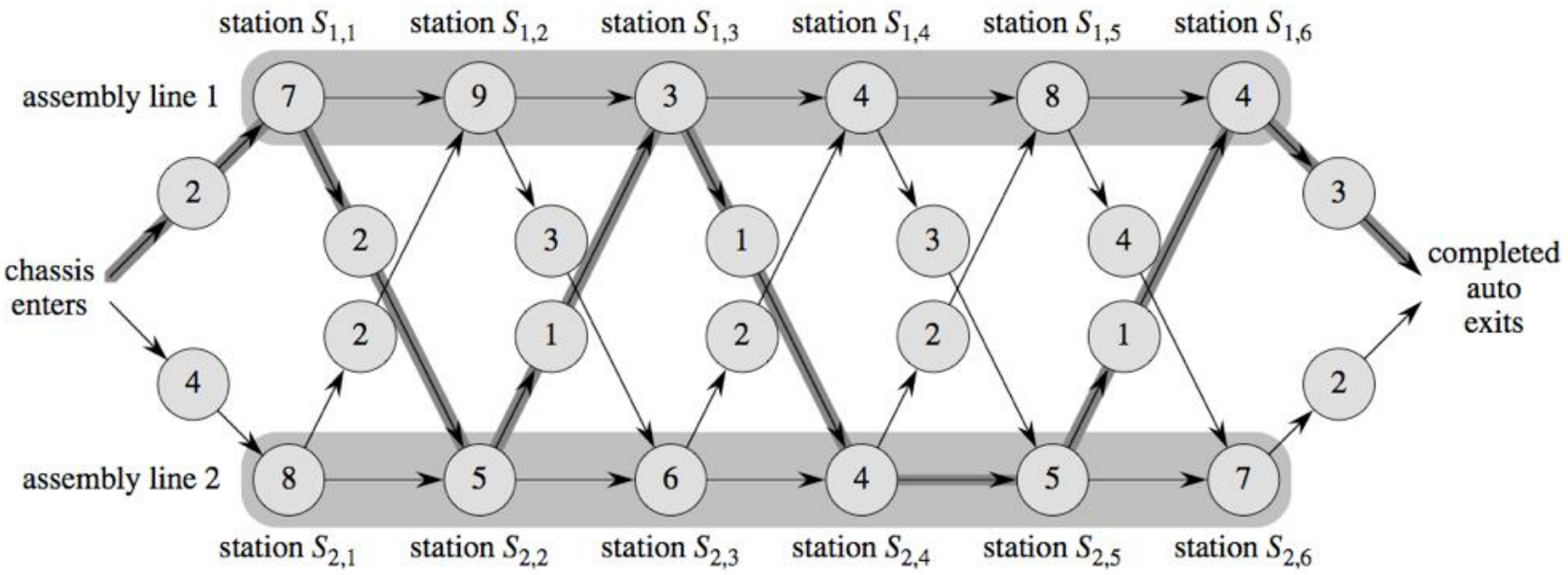
Linhas de produção

Objetivo: encontrar o menor tempo para fabricação de uma peça em uma fabrica com 2 linhas de produção



Linhas de produção

Há um tempo pré-determinado para executar uma operação e para mudar de linha



Linhas de produção

Caso base:

- Entrada da linha

Casos gerais:

- Mudar de linha ou manter-se na mesma?

Linhas de produção

Recuperação do caminho:

```
i ← l*
print "linha 'i' estação 'n-1' "
for j ← n-1 downto 1
        i ← l[i][j]
        print "linha 'i' estação 'j-1' "
```

Via: Crônicas de Wesley