Optimization in the Loop

Implementing and Testing Scheduling Algorithms with SimuLTE

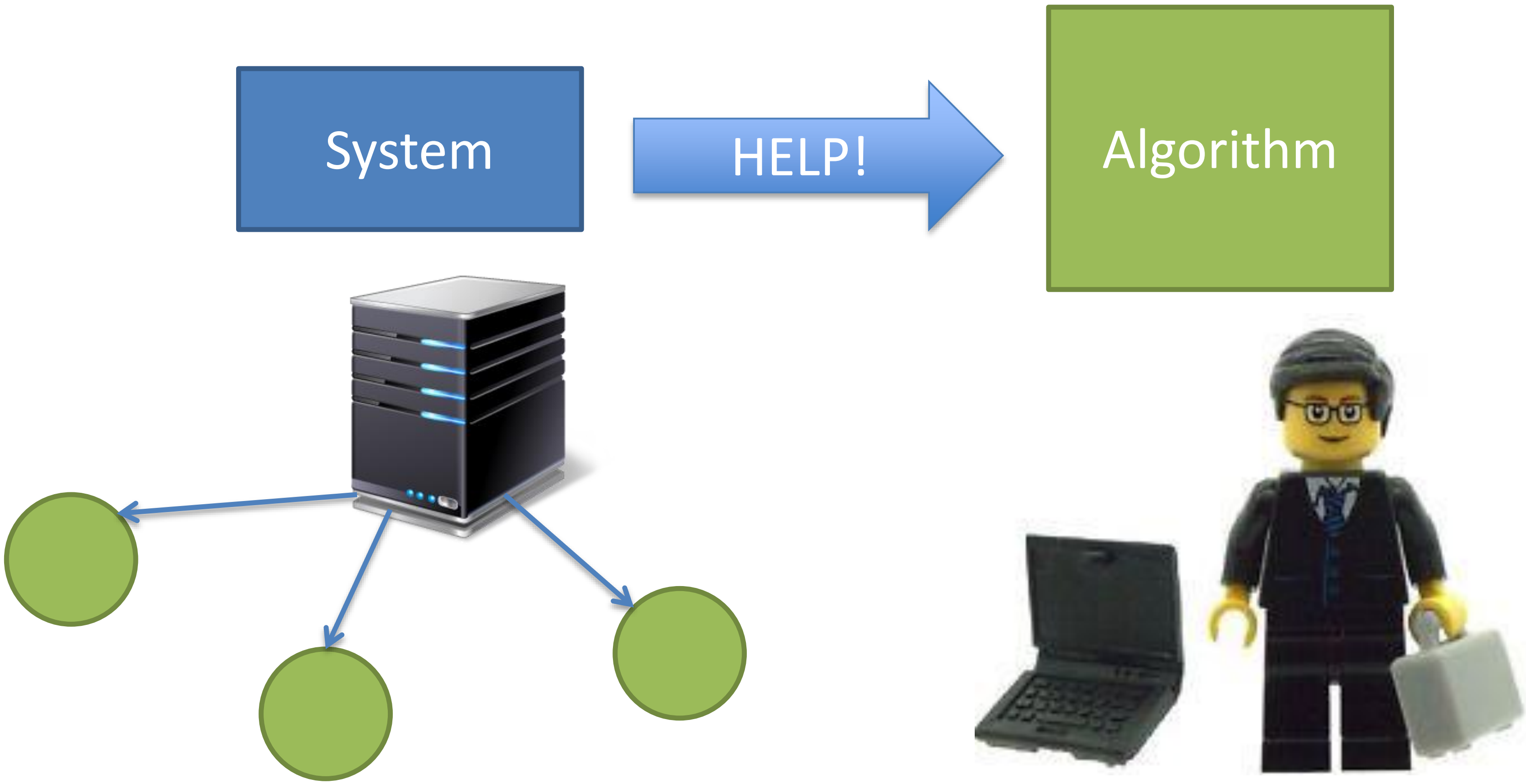
Antonio Virdis
University of Pisa

- Prof. Giovanni Stea
- Giovanni Nardini

Outline

- Why Optimization
- Going into the Loop
- Methods
- Example

An everyday problem



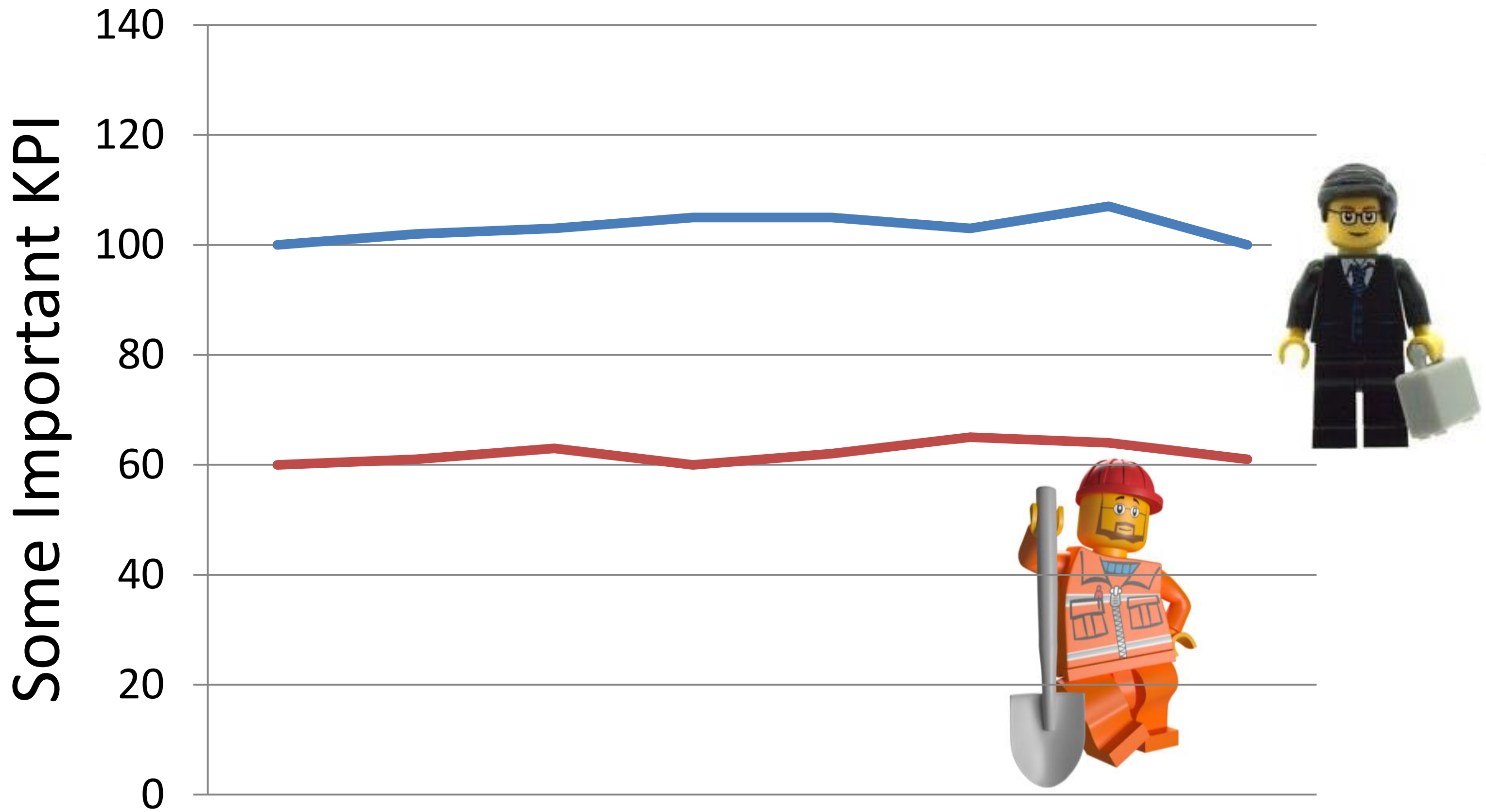
Comparing Results

system

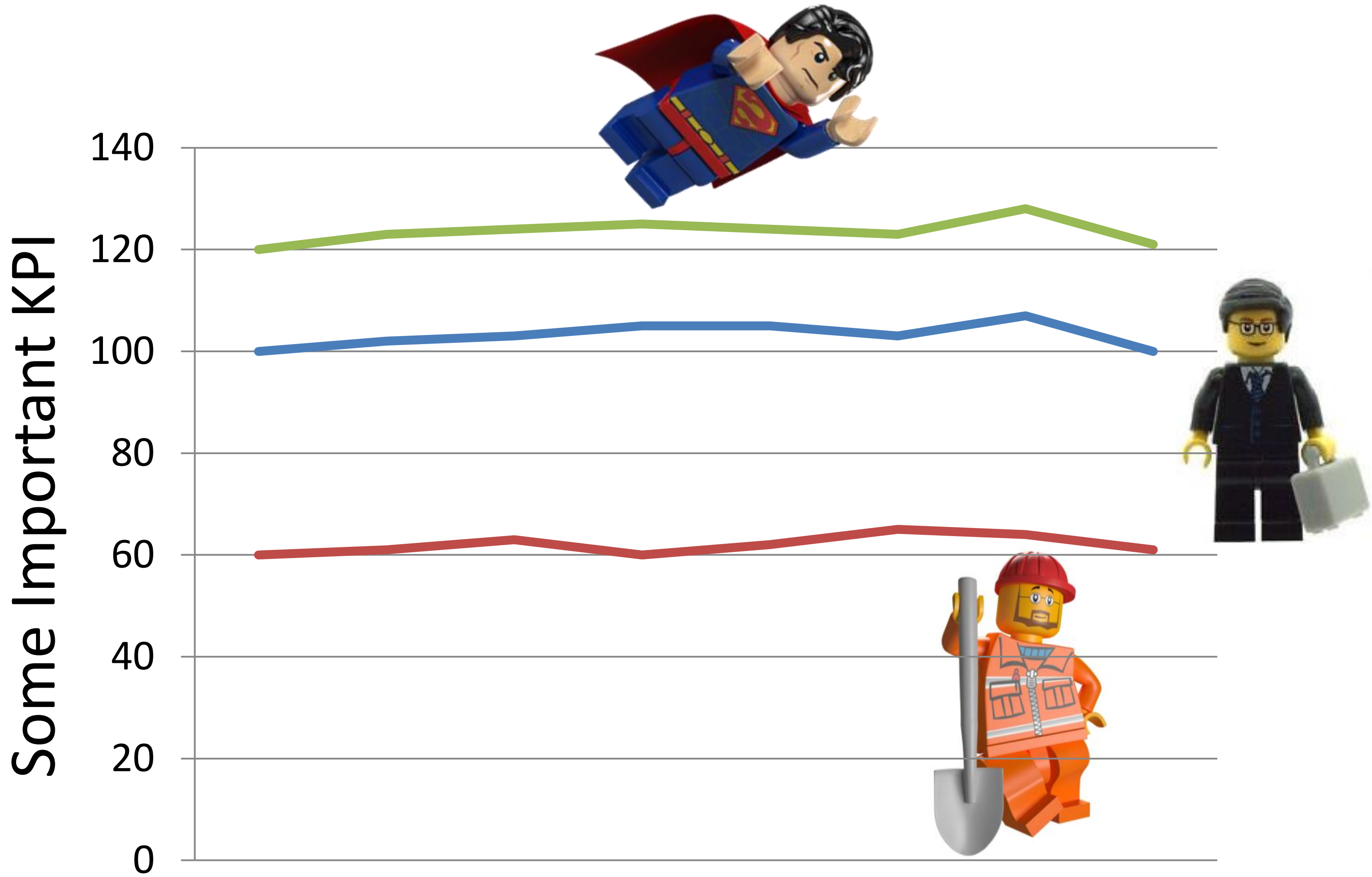
Algorithm 1

Algorithm 2

Comparing with the best



Comparing with the best



Comparing Results

~~I'm better than the optimum~~

system

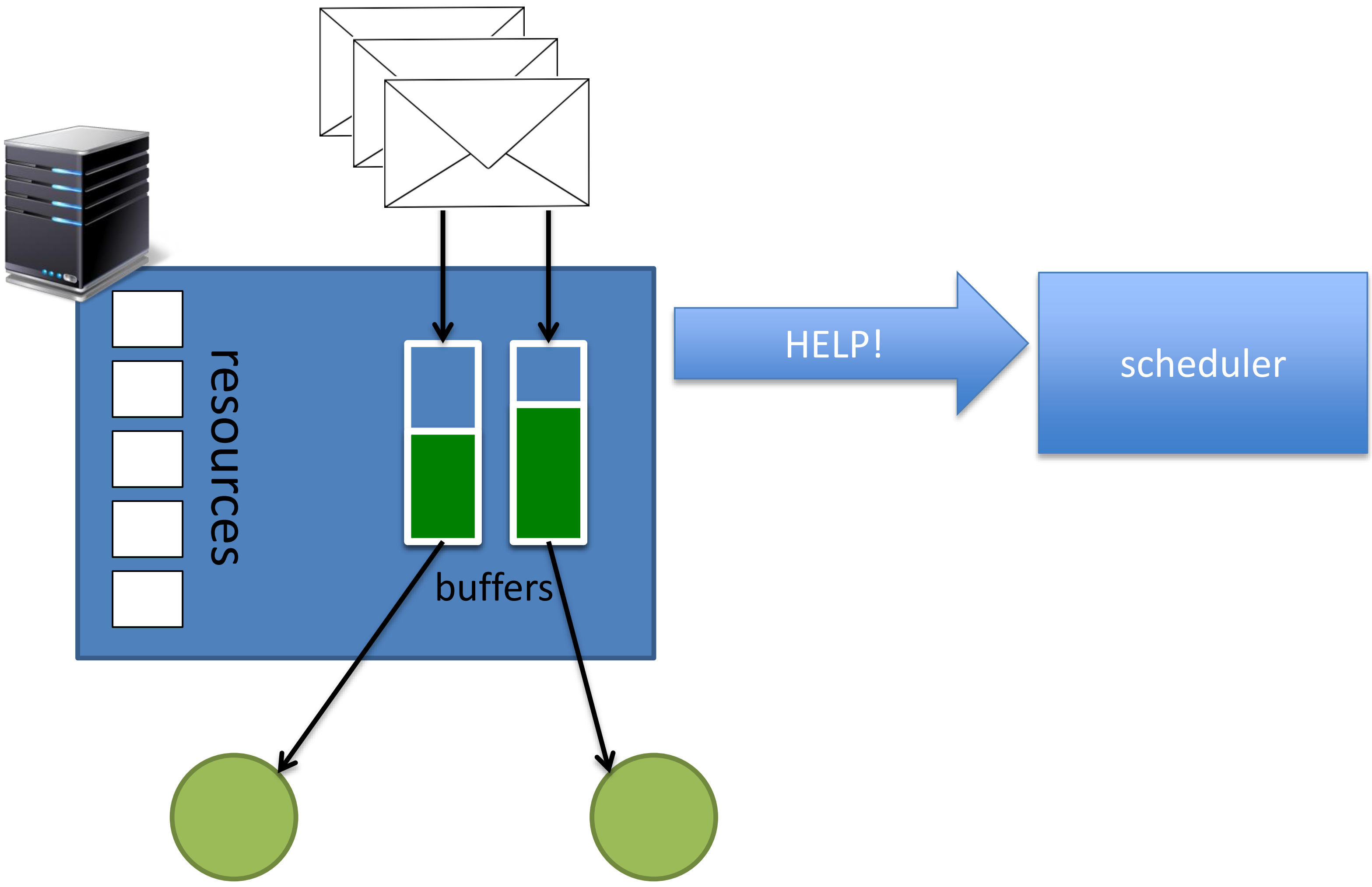
$$\max \sum_i a \times x_i$$

$s.t.$

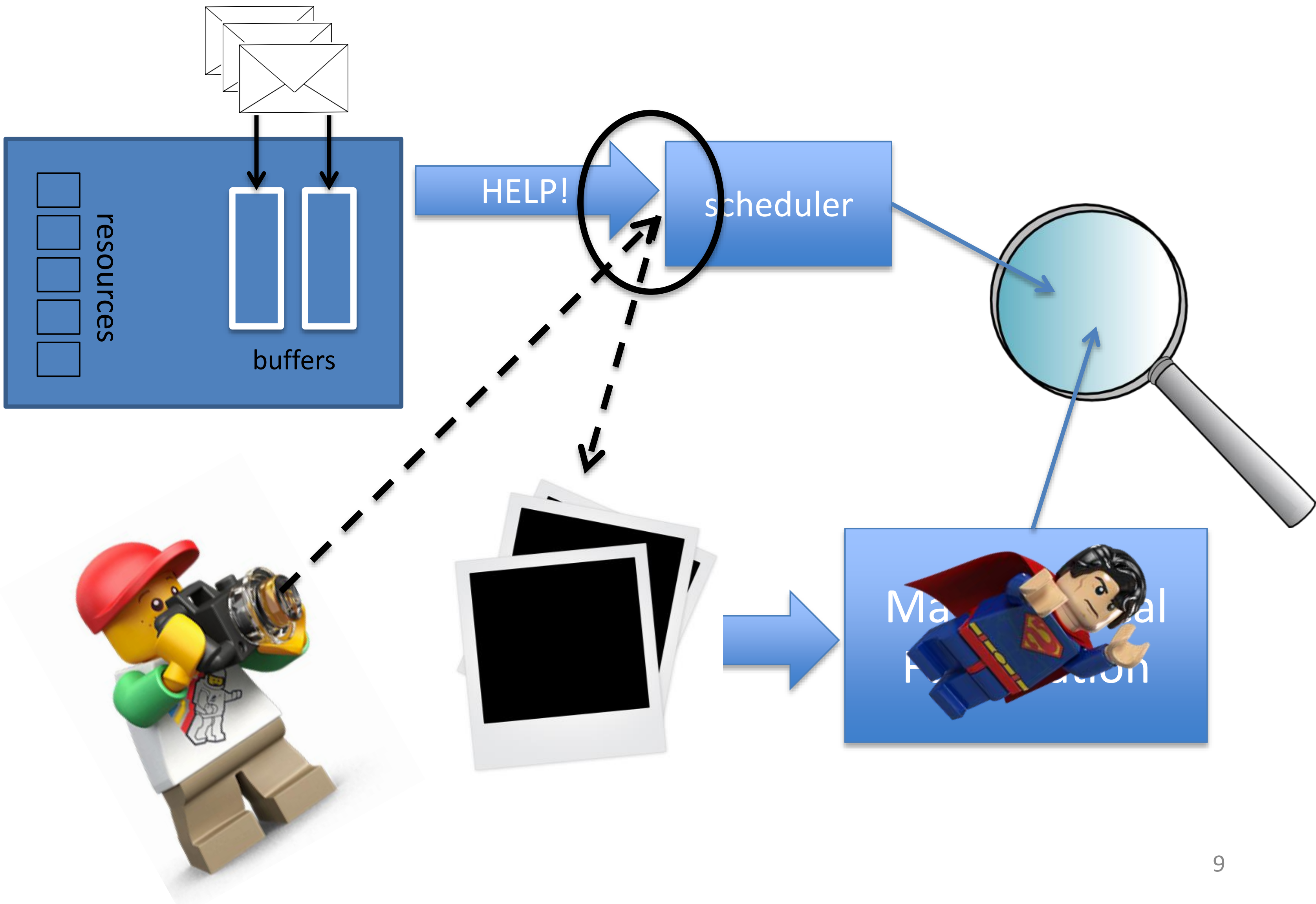
$$x_i + p_i \leq M$$

…

A simple problem



Taking a Photo



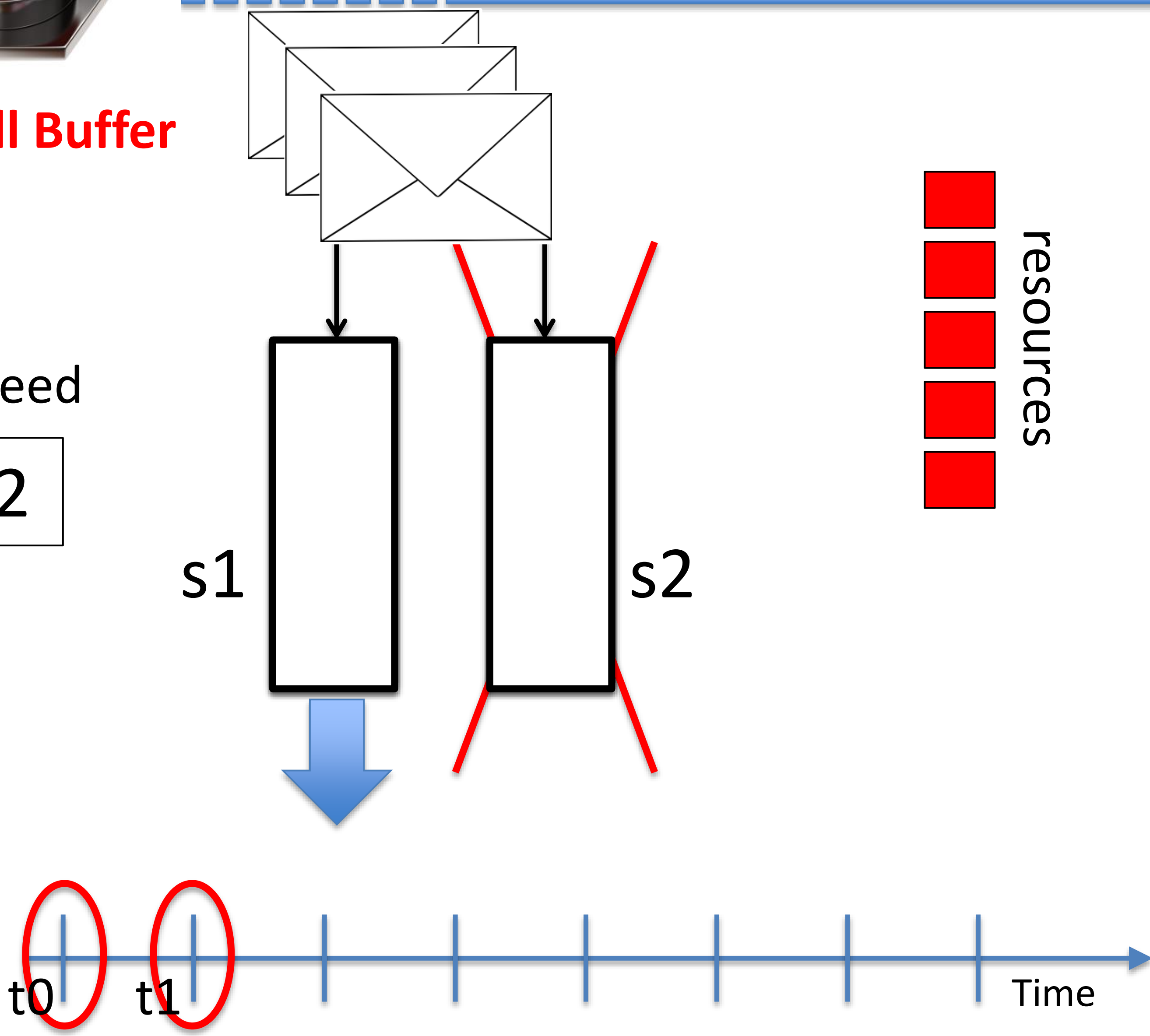
Quiz 1

Quiz 1

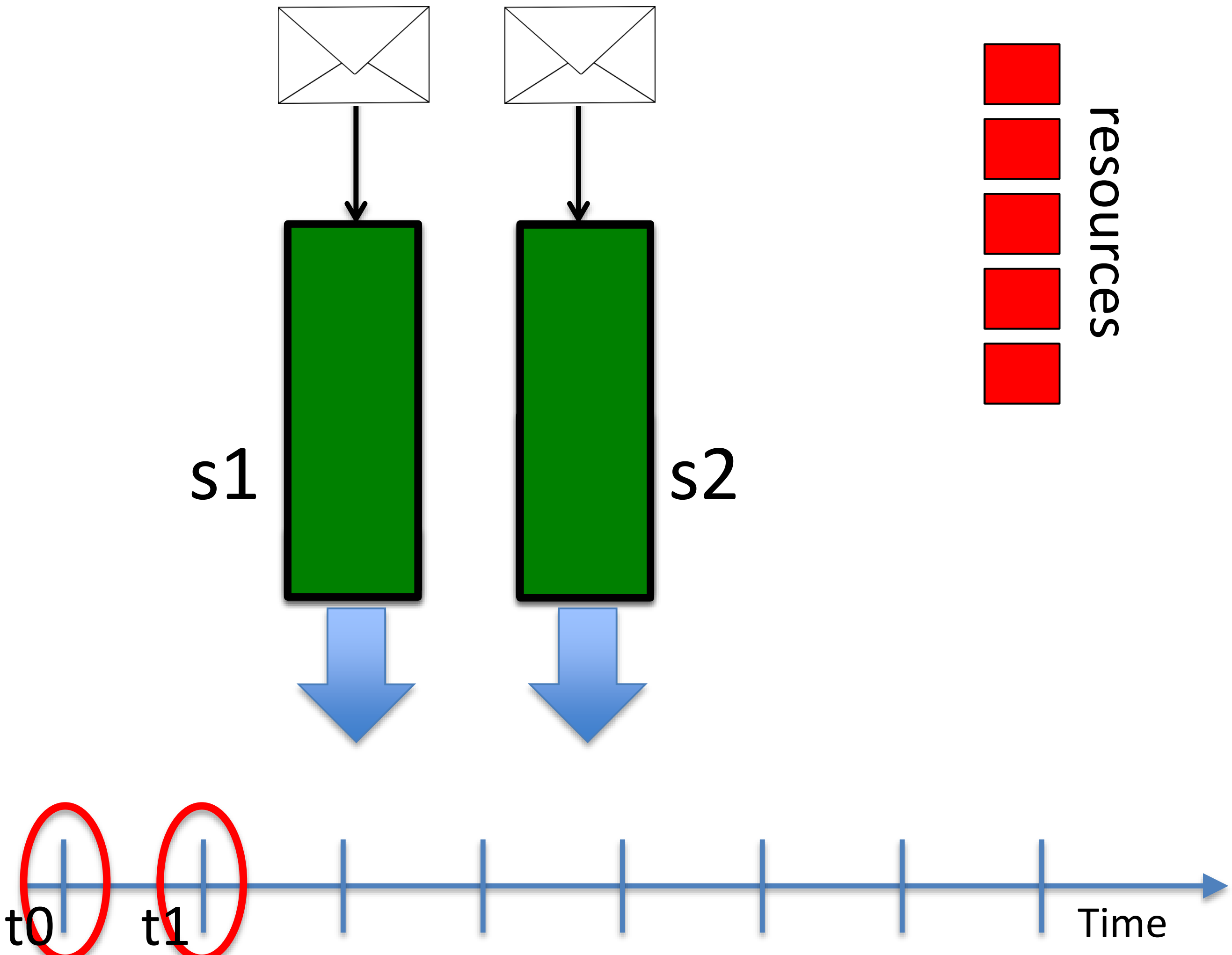
Full Buffer

output speed

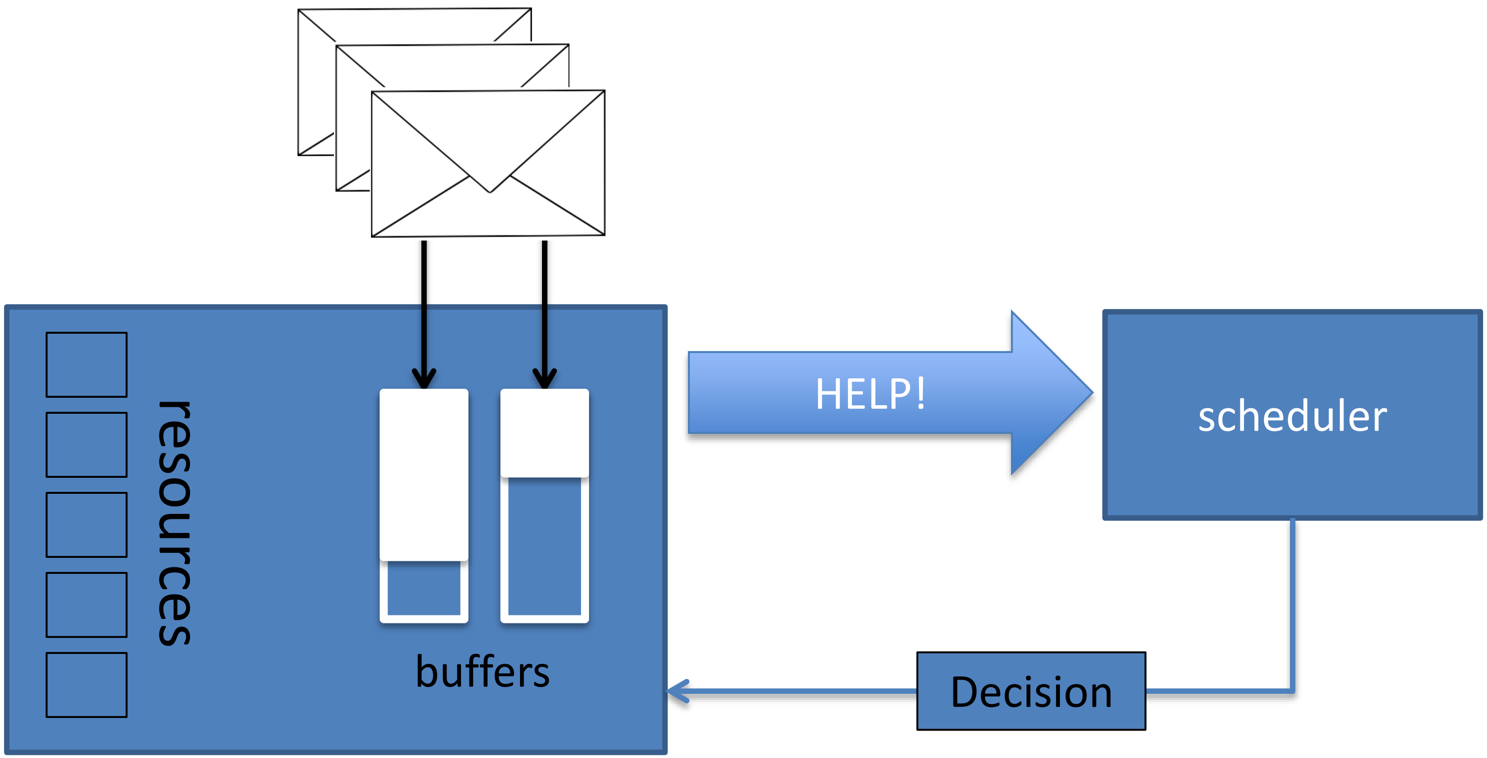
$s1 > s2$



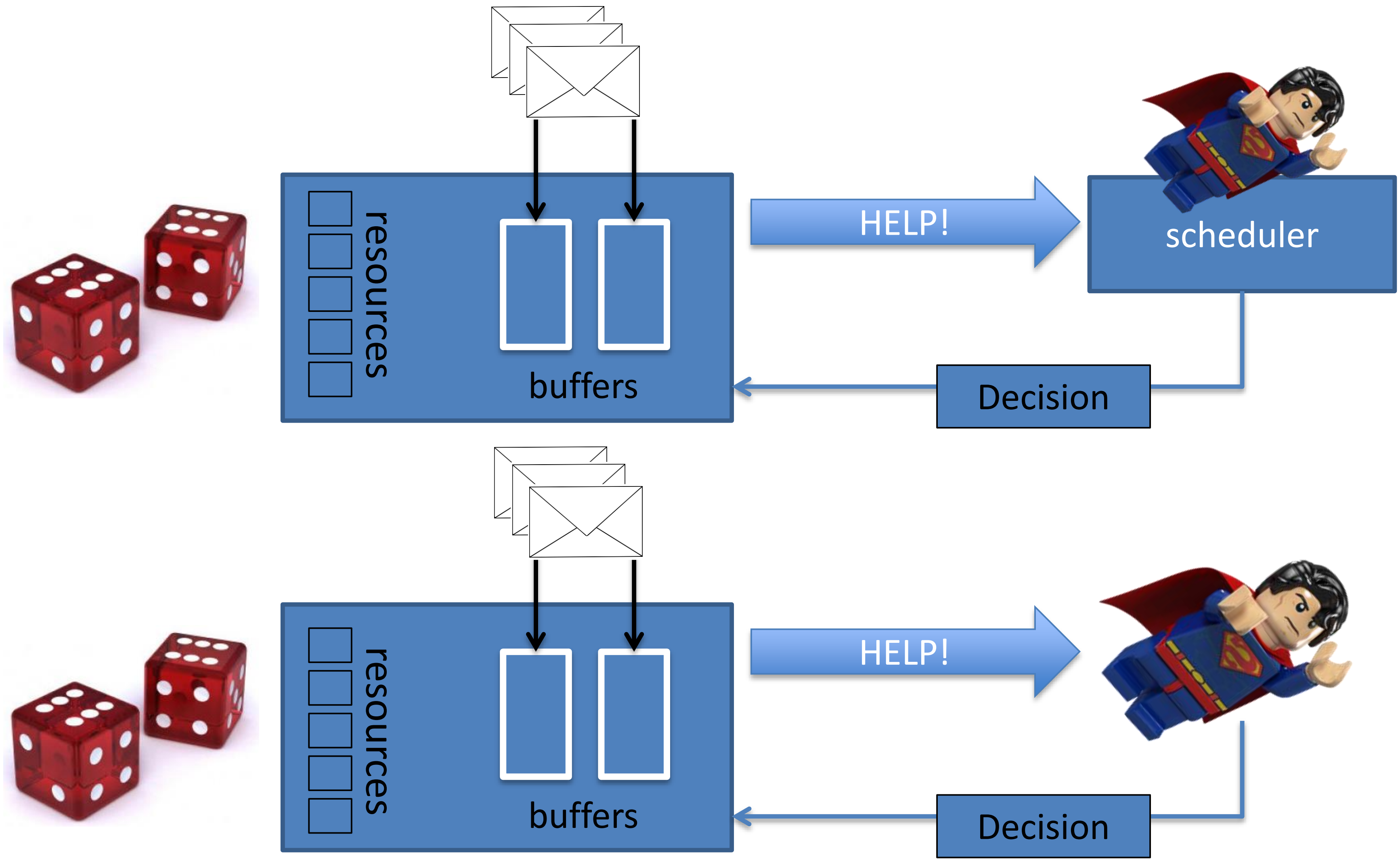
Finite Buffer: CBR



How does the system evolve



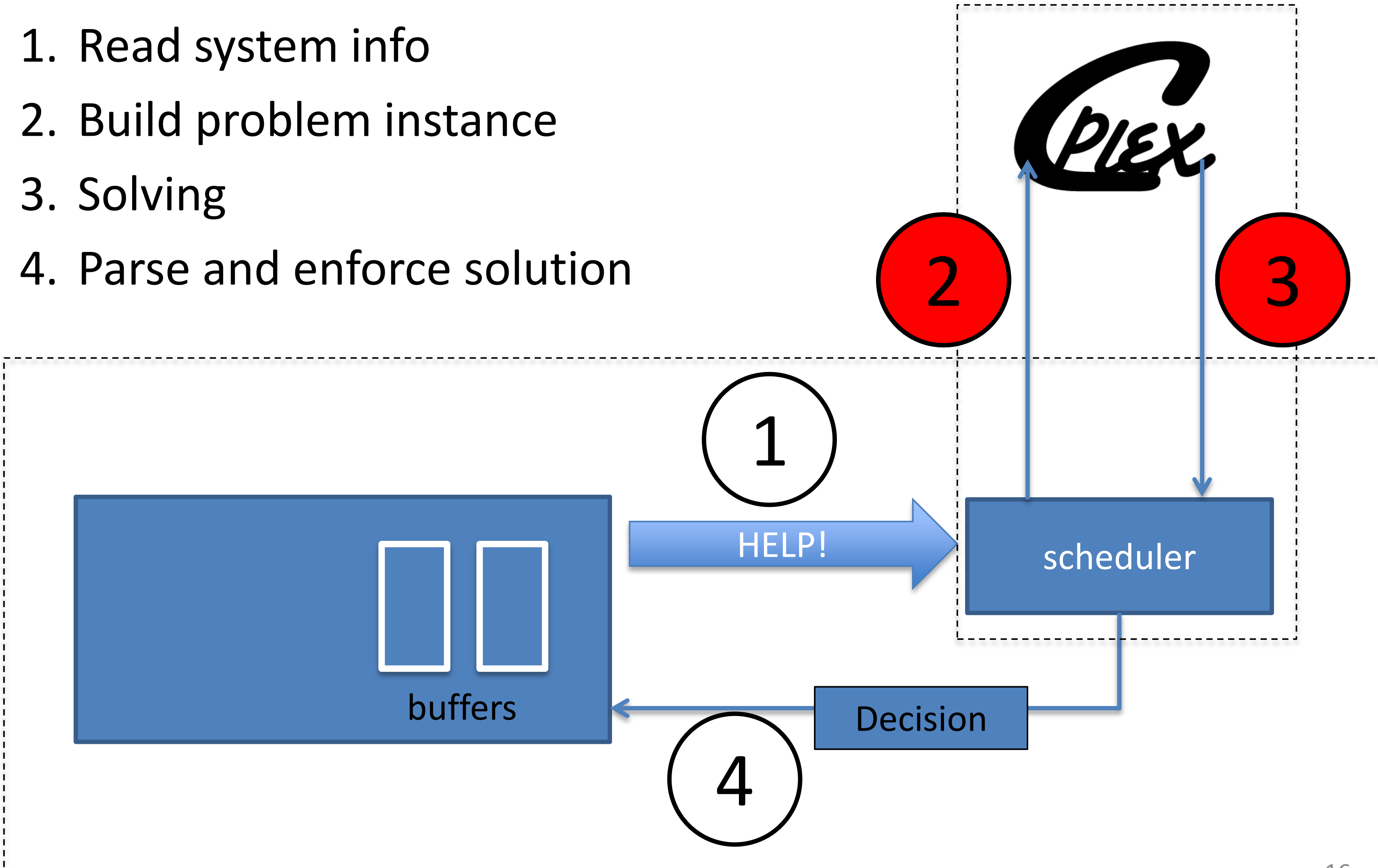
From outside to inside



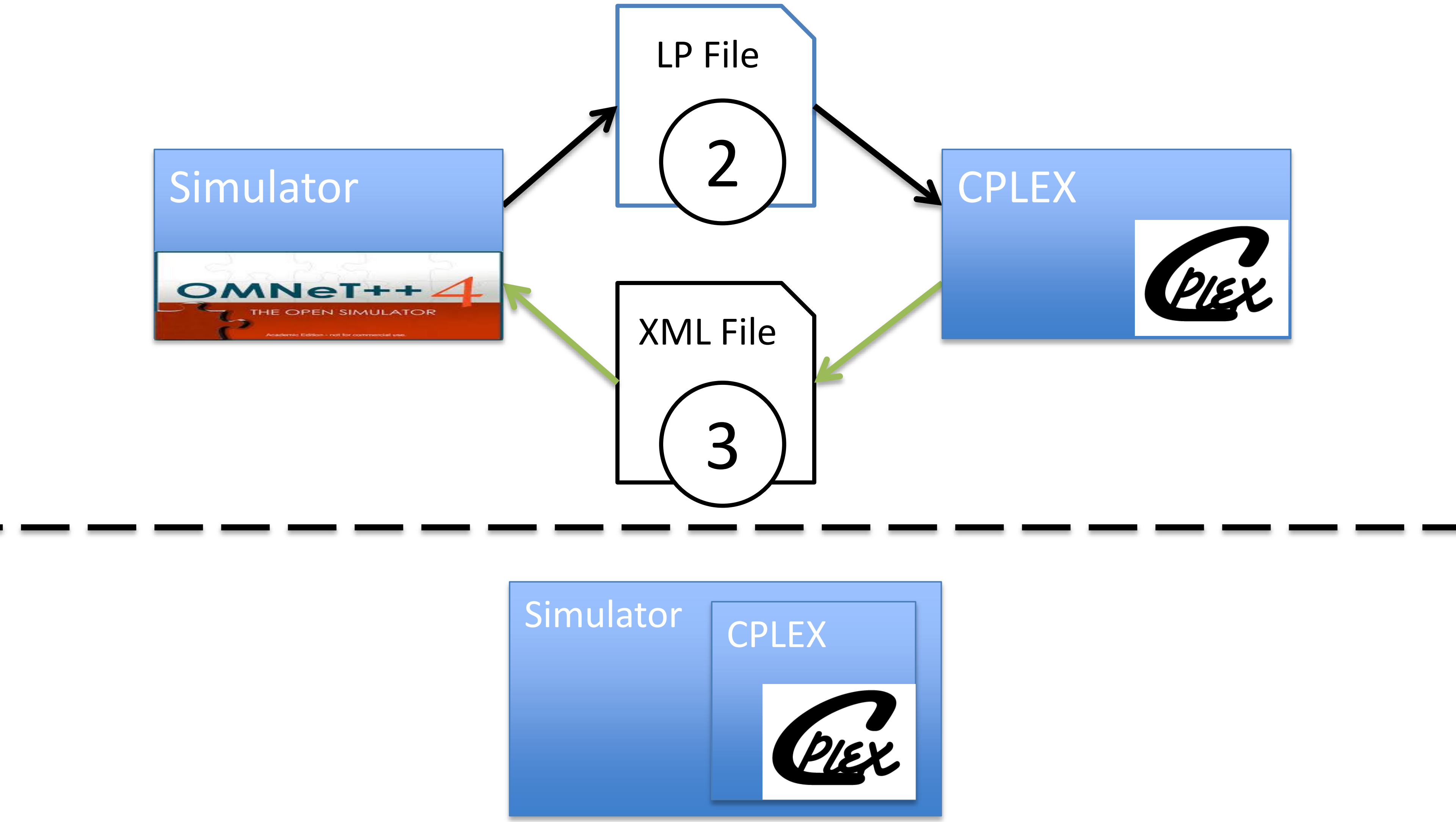
Going Into The Loop

Overview

1. Read system info
2. Build problem instance
3. Solving
4. Parse and enforce solution



2 methods



LP File
2
Simulator
OMNeT++ 4
THE OPEN SIMULATOR
CPLEX
XML File
3
Simulator
CPLEX

LP file

② Building A problem File

$$\max \sum_{i}^{N} x_i$$

```
for( i=0 ; i<N ; ++i )
    stream << "x" << i << " + ";
```

$$s.t.$$

$$x_i + p_i \le M \quad \forall i$$

```
for( i=0 ; i<N ; ++i )
    ...............
```

...

XML
file

3 Reading Results

– XML Management

- Built-in in OMNeT
- Easy to implement manually

```xml
<variable name="x0" index="0" value="51"/>
<variable name="x1" index="1" value="0"/>
<variable name="x2" index="2" value="141"/>
<variable name="x3" index="3" value="0"/>
<variable name="p0" index="4" value="141"/>
<variable name="p1" index="5" value="0"/>
```

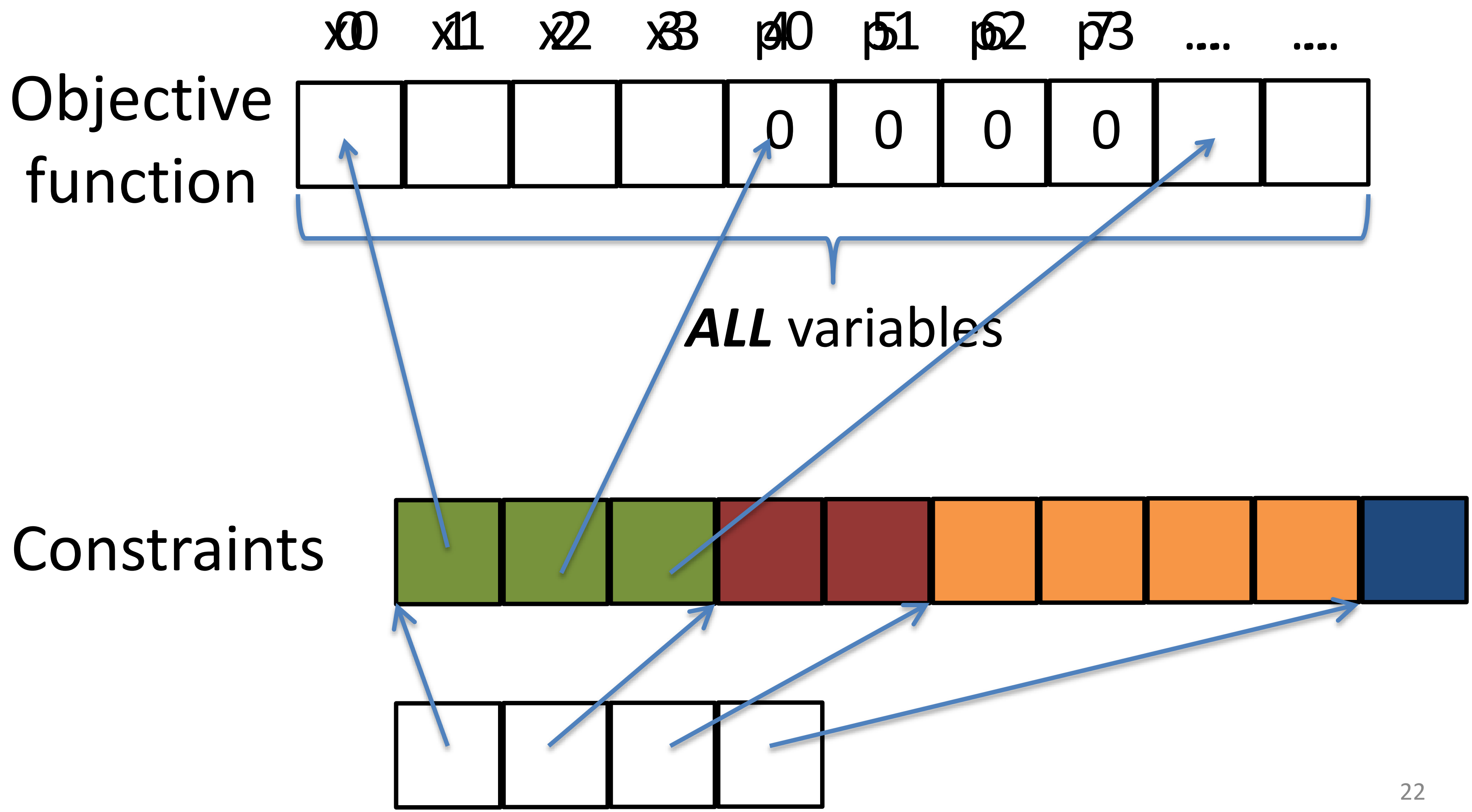
2° method: API

- Idea: can we use CPLEX as an API?
  - ***Callable Library***: matrix-based C-written API
  - ***Concert Tecnology***: a set of modeling objects (also) in C++

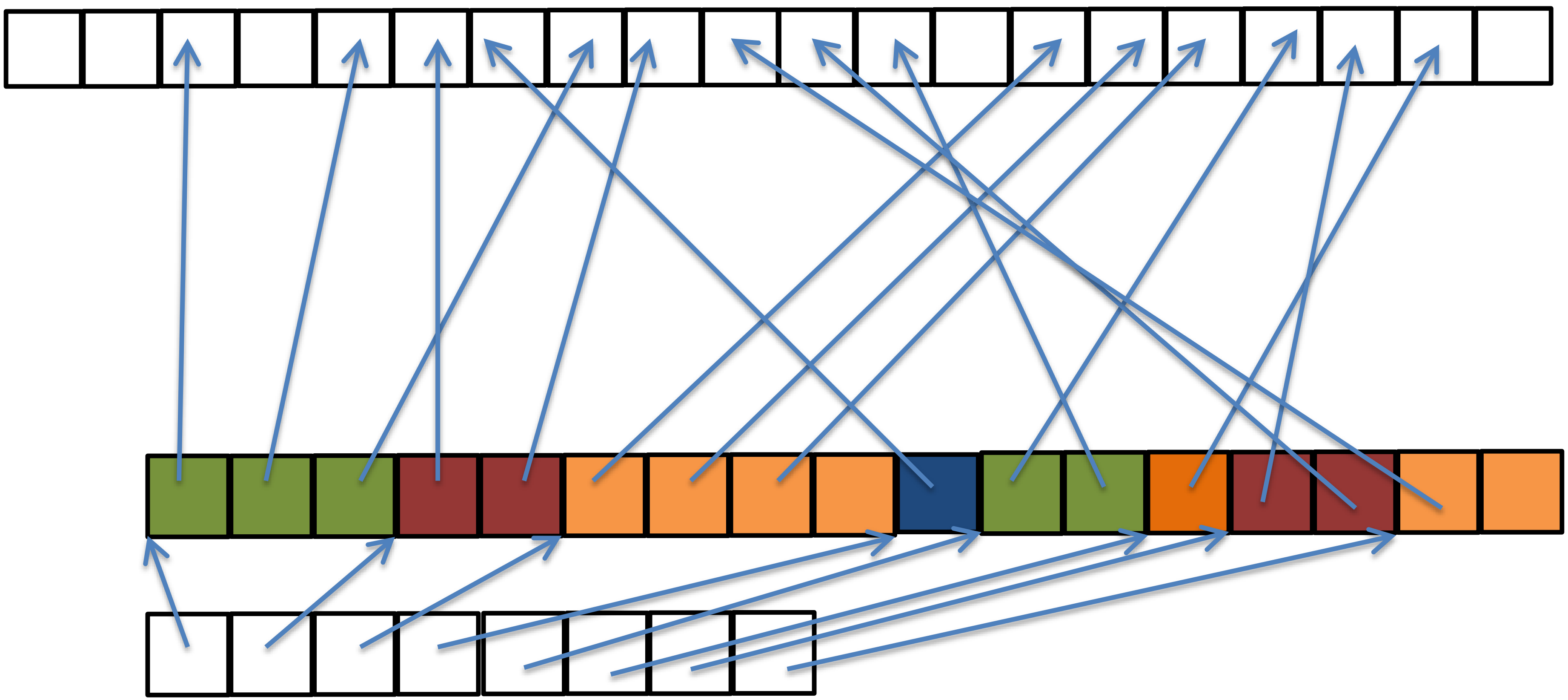
Including CPLEX

- **TELL OMNET**:
  - where the .h files are located
  - where the dynamic libraries are located
  - wich dynamic library to include
  - enable the I_STD preprocessor macro
- Can be done via the ***Project Properties*** of OMNeT++

Matrix representation



Matrix representation



Custom C++ Interface

- Generally variables are in the form:
  - X_i One pedi**x**
  - $y_{i,j}$ Two pedi**ces**

$$y_{i,j}$$

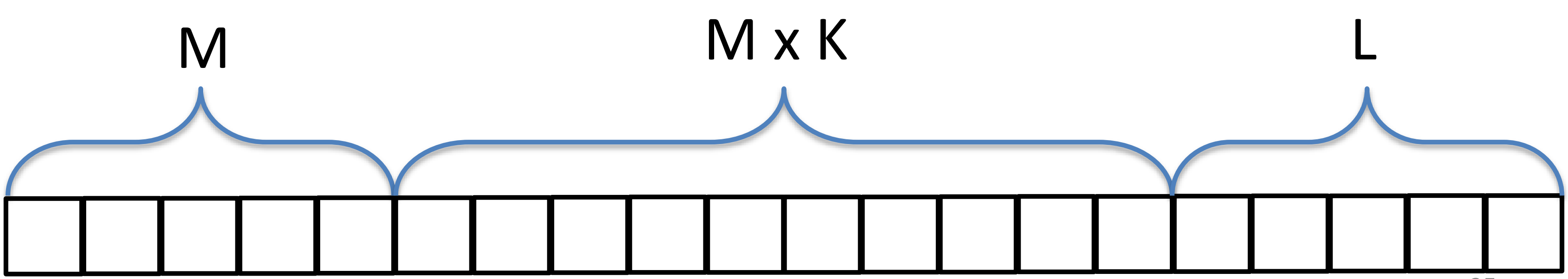
$$\left.\begin{array}{l} 0 \leq i < M \\ \\ 0 \leq j < K \end{array}\right\} M \times K$$

2° Method: variables

Name , #1st , #2nd

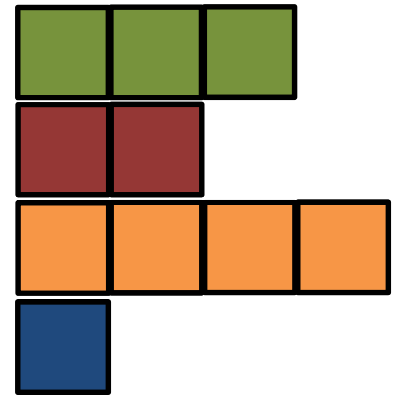
x_i	M	0
$y_{i,j}$	M	K
z_n	L	0

Access with ***local indexes***

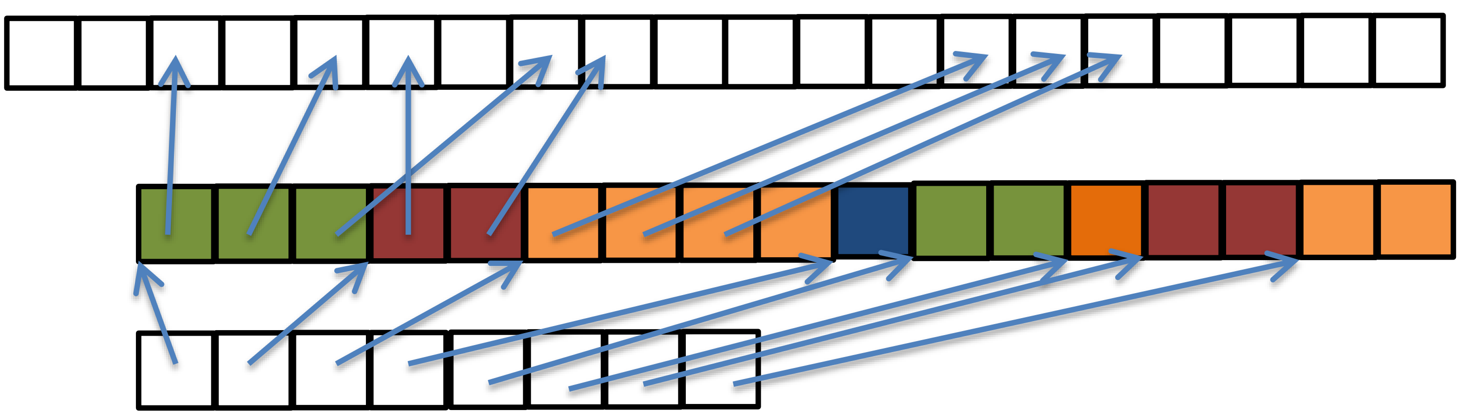


2° Method: constraints

- Add constraints one by one usign ***local indexes***



- Build the problem at the end **one-shot**



Reading The Output

XML

```xml
<variable name="x0" index="0" value="51"/>
<variable name="x1" index="1" value="0"/>
<variable name="x2" index="2" value="141"/>
<variable name="x3" index="3" value="0"/>
<variable name="p0" index="4" value="141"/>
<variable name="p1" index="5" value="0"/>
```

index 0 1 2 3 4 ....

solution

Quiz 2:

x_i Î $\{0,1\}$ Binary values

$$x[i] \stackrel{?}{==} 1$$

Quiz 2:

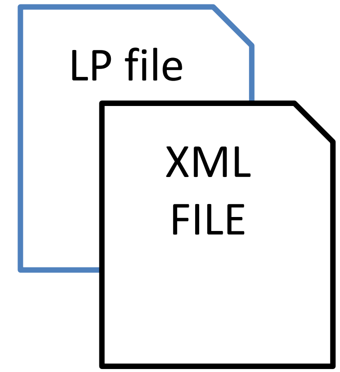
```
x0  -> 0
x1  -> 0
x2  -> 1
x3  ->
x4  ->
x5  ->
x6  ->
x7  ->
x8  -> 1
x9  -> 1
x10 -> 1
x11 -> 0
```

Do not trust **double** values

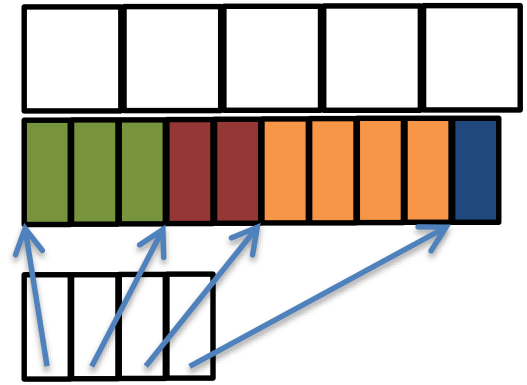
```
x0  -> 0.00000000
x1  -> 0.00000000
x2  -> 1.00000000
       00000000
       00000000
       00000000
       00000000
       00000000
x8  -> 1.00000001
x9  -> 1.00000000
x10 -> 1.00000000
x11 -> 0.00000000
```

Pros and Cons

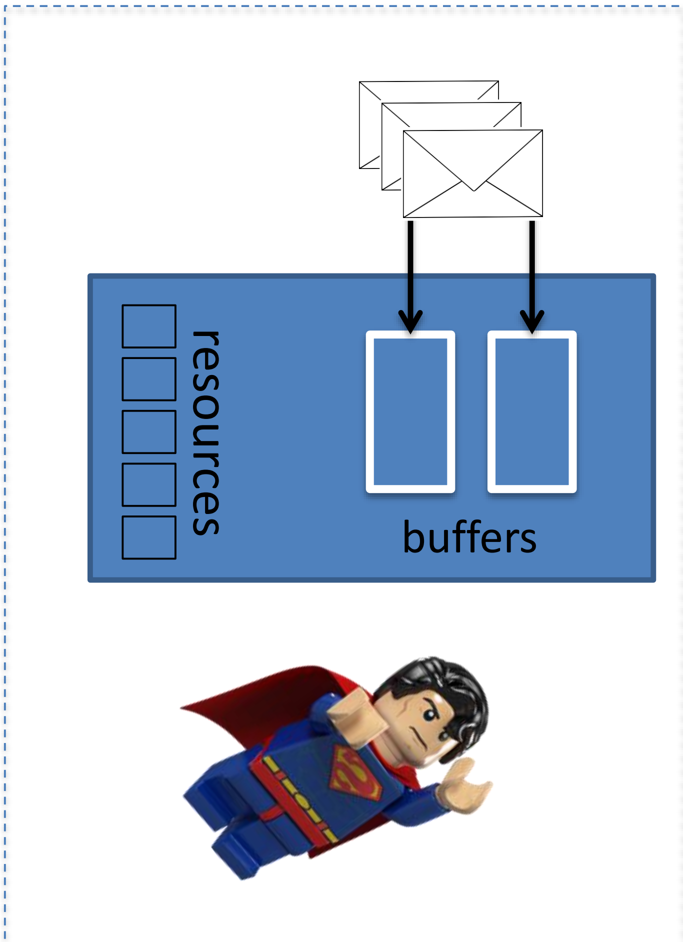
- Easy to build
- Generally slower



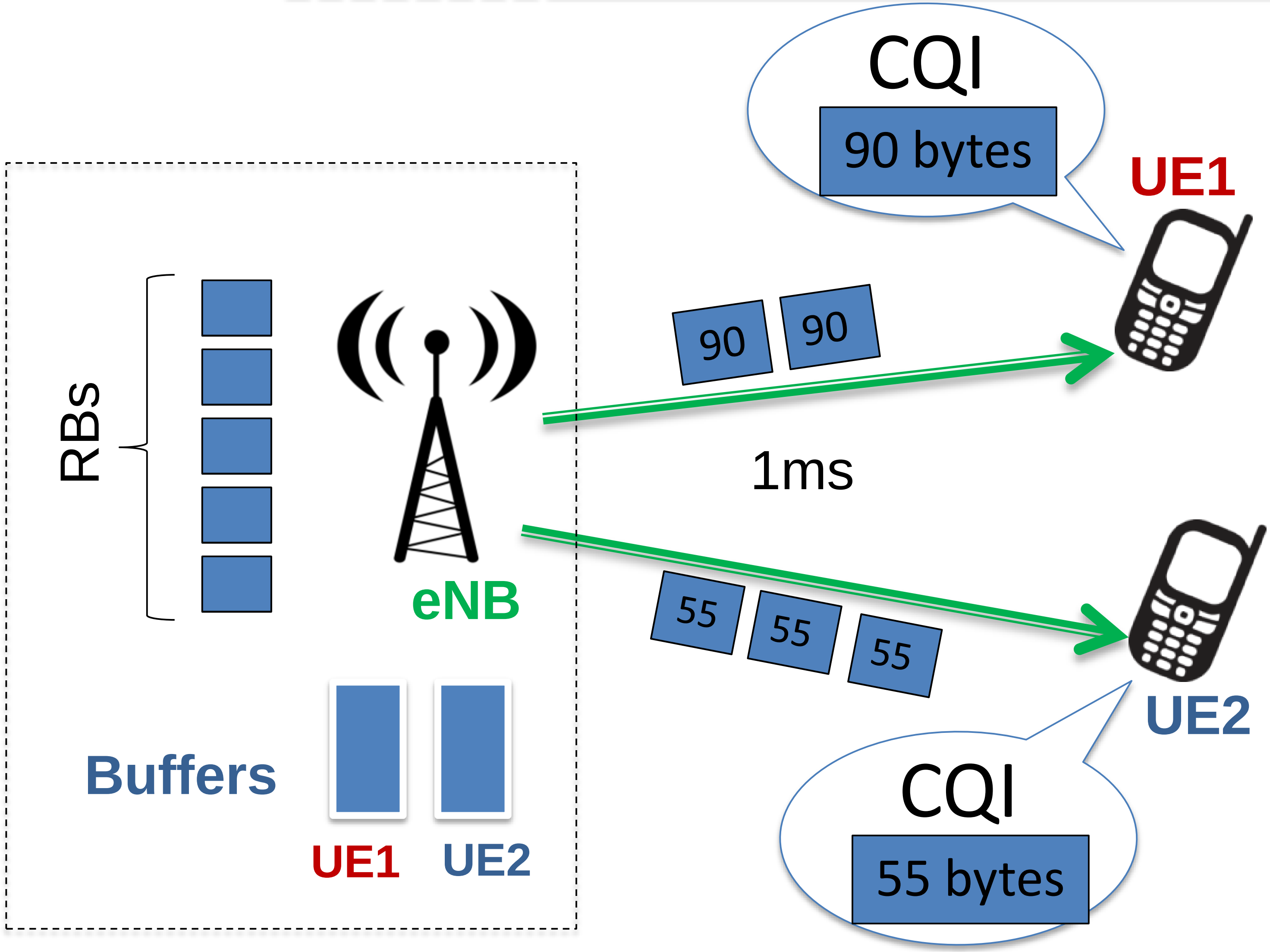
- Generally faster
- Requires API knowledge



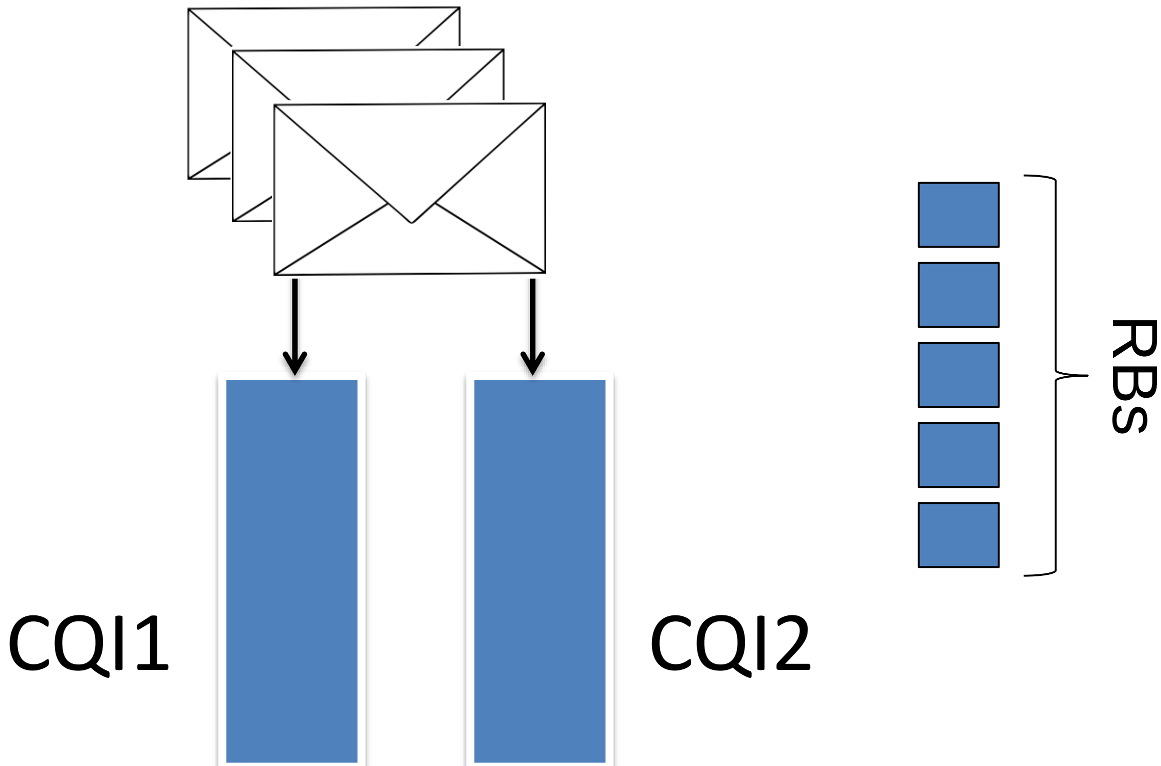
Optimization in SimuLTE



LTE

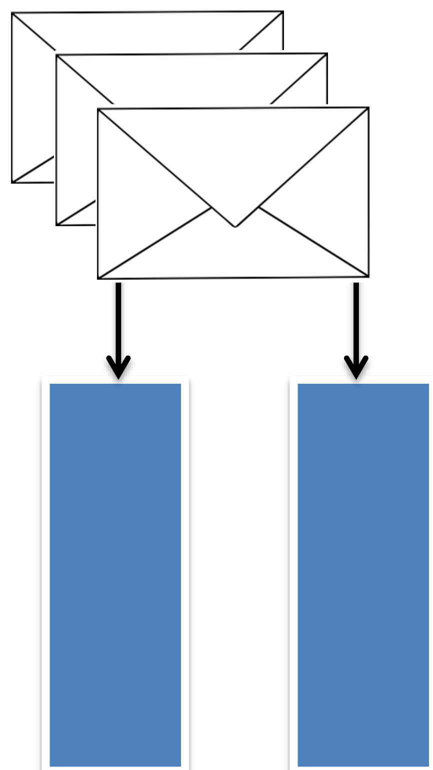

CQI
90 bytes
UE1
RBs
90
90
1ms
eNB
55
55
55
UE2
Buffers
UE1
UE2
CQI
55 bytes

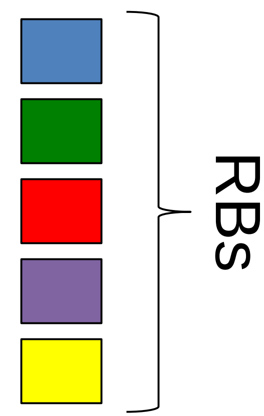
Resource allocation in LTE



Allocate **RBs** to **UEs**

Multi Band Scheduling

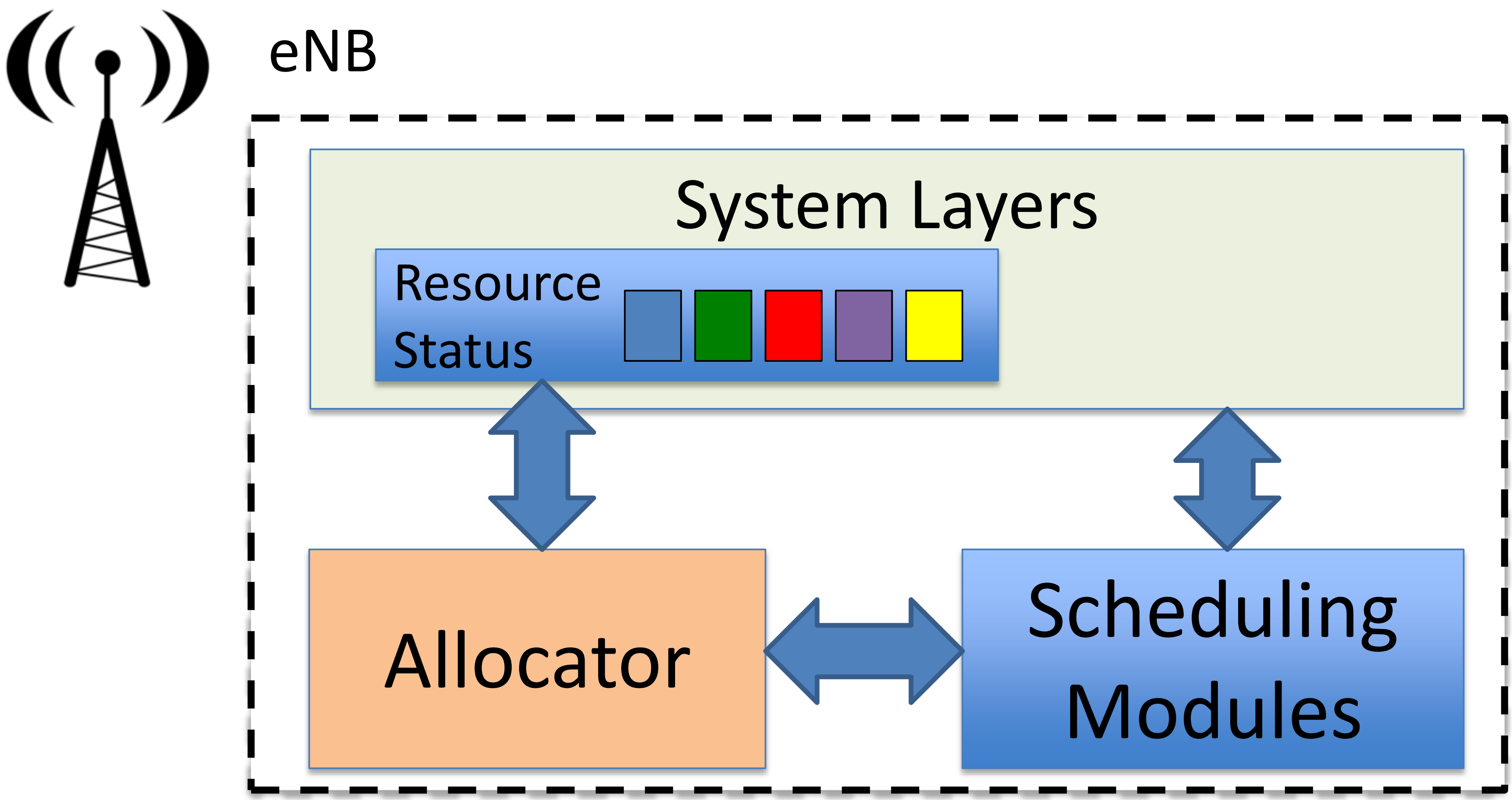




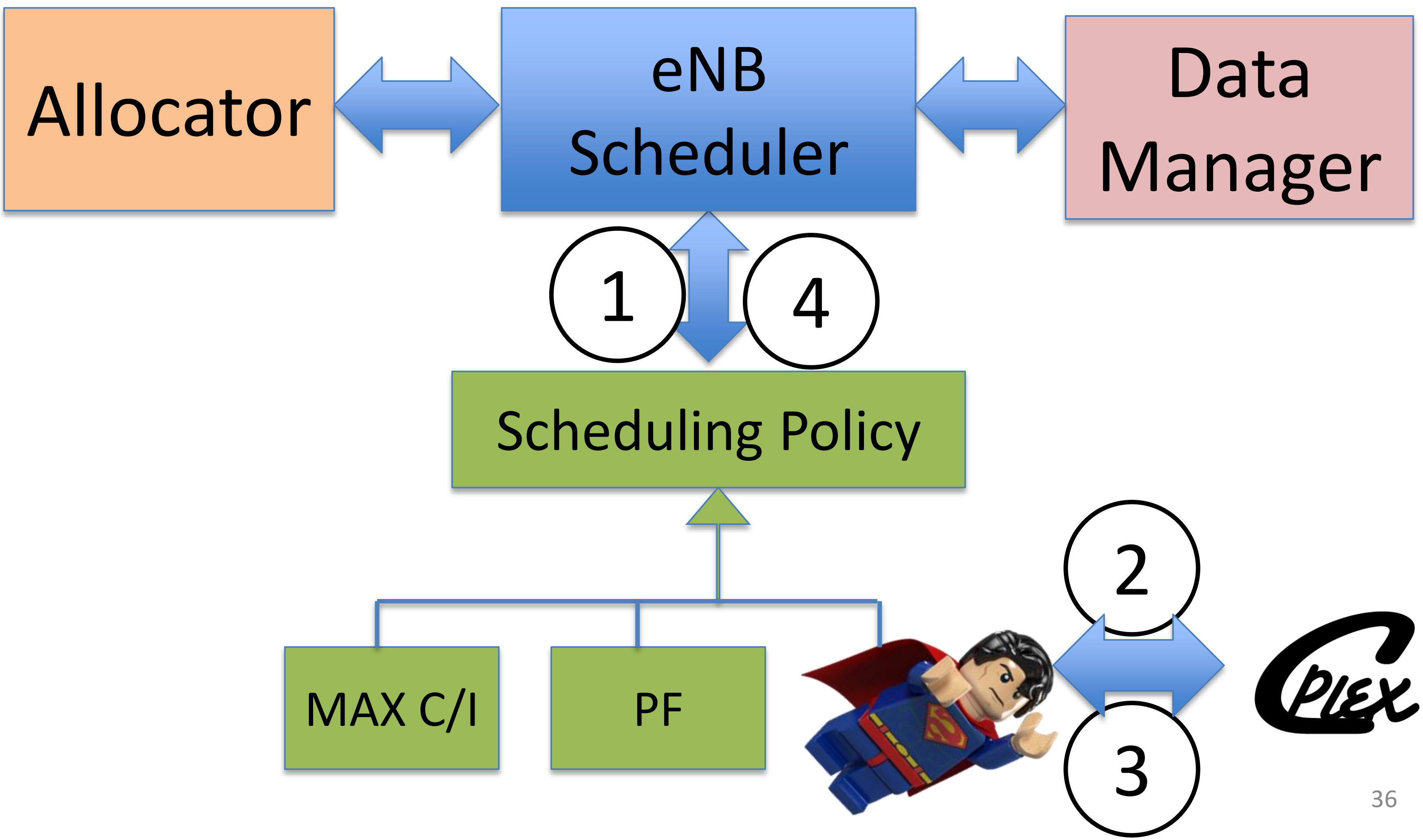
CQI1 [0]
CQI1 [1]
CQI1 [2]
…

CQI2 [0]
CQI2 [1]
CQI2 [2]
…

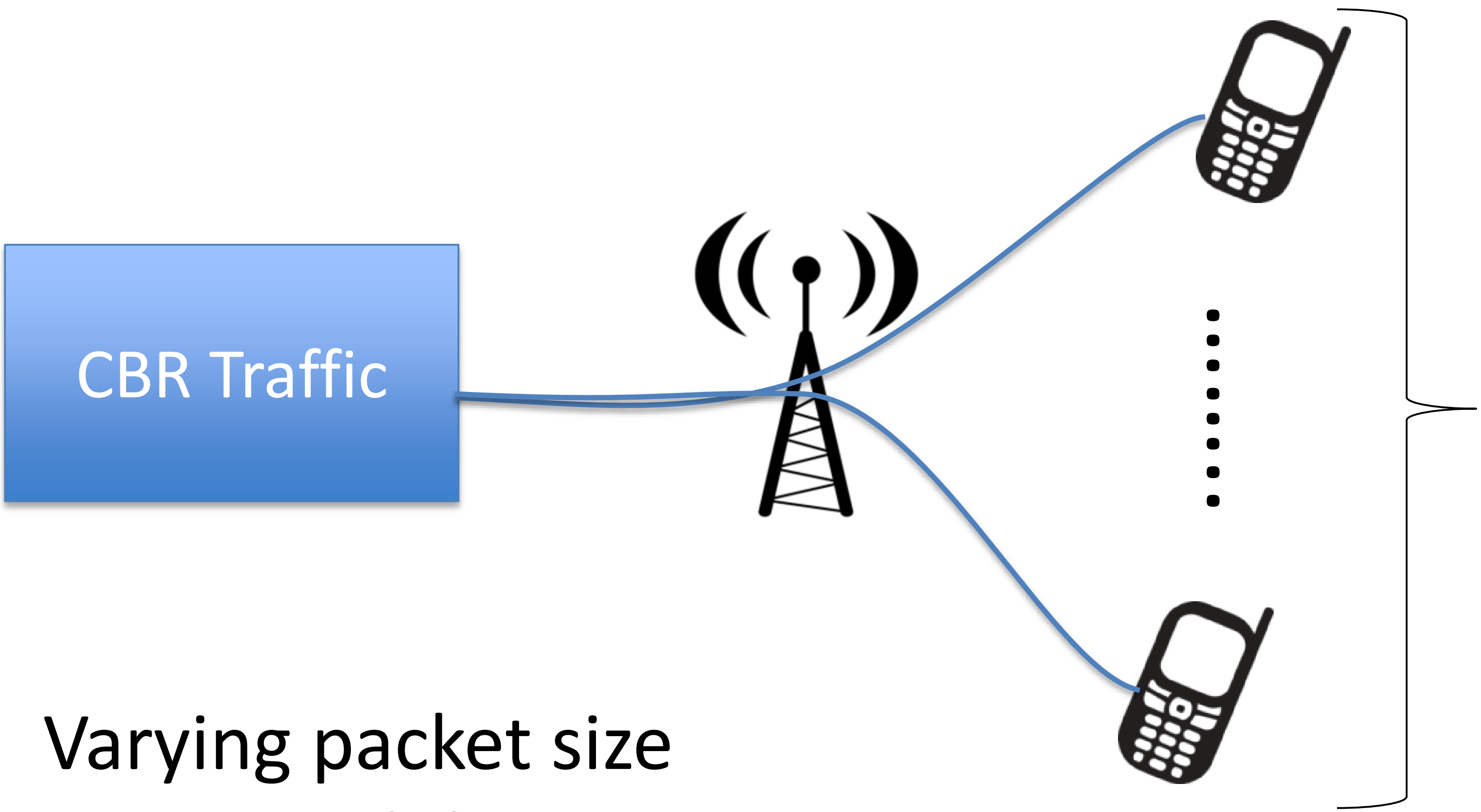
SimuLTE: Scheduling structure



SimuLTE: Scheduler Hierarchy

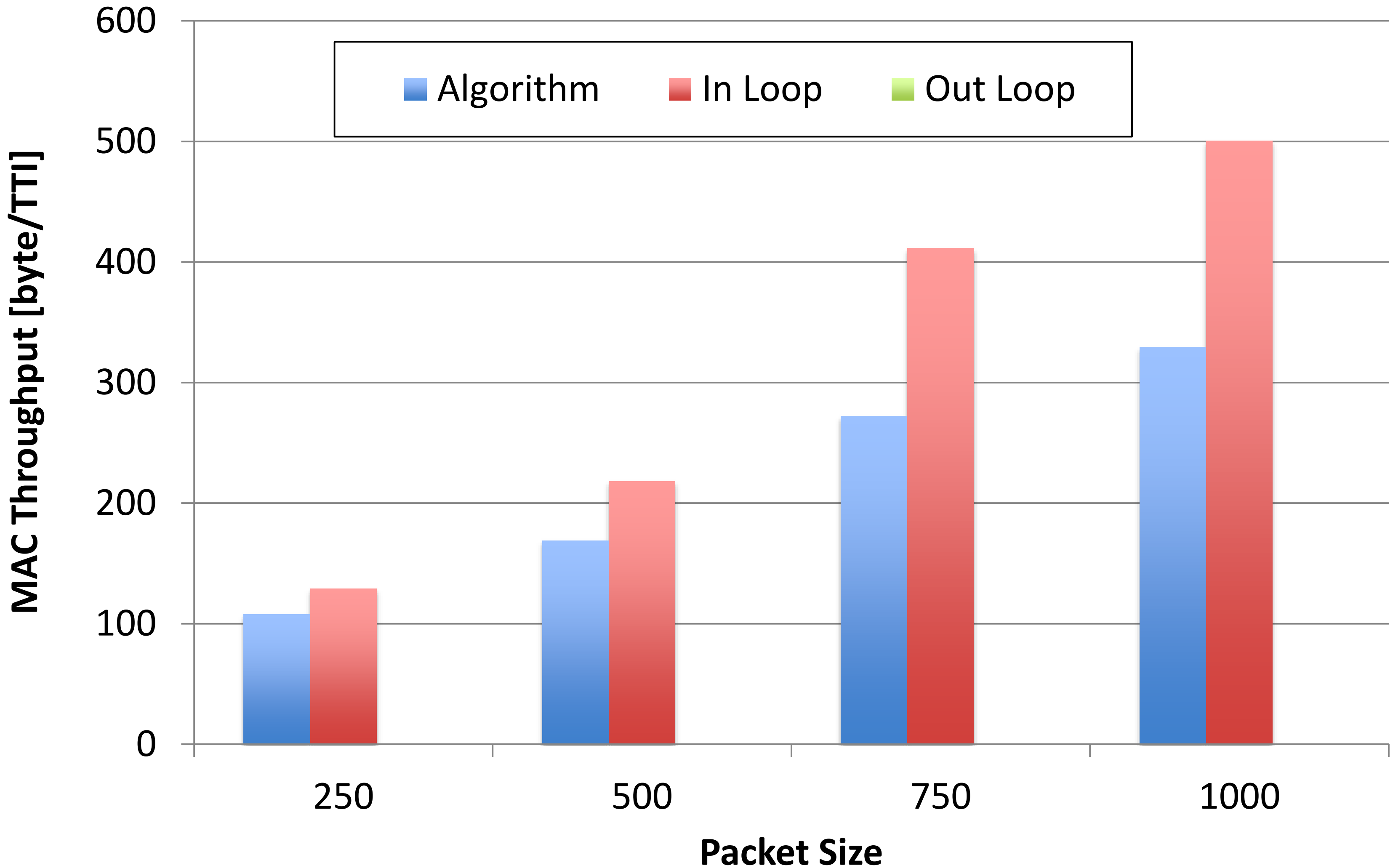


Simulation Scenario

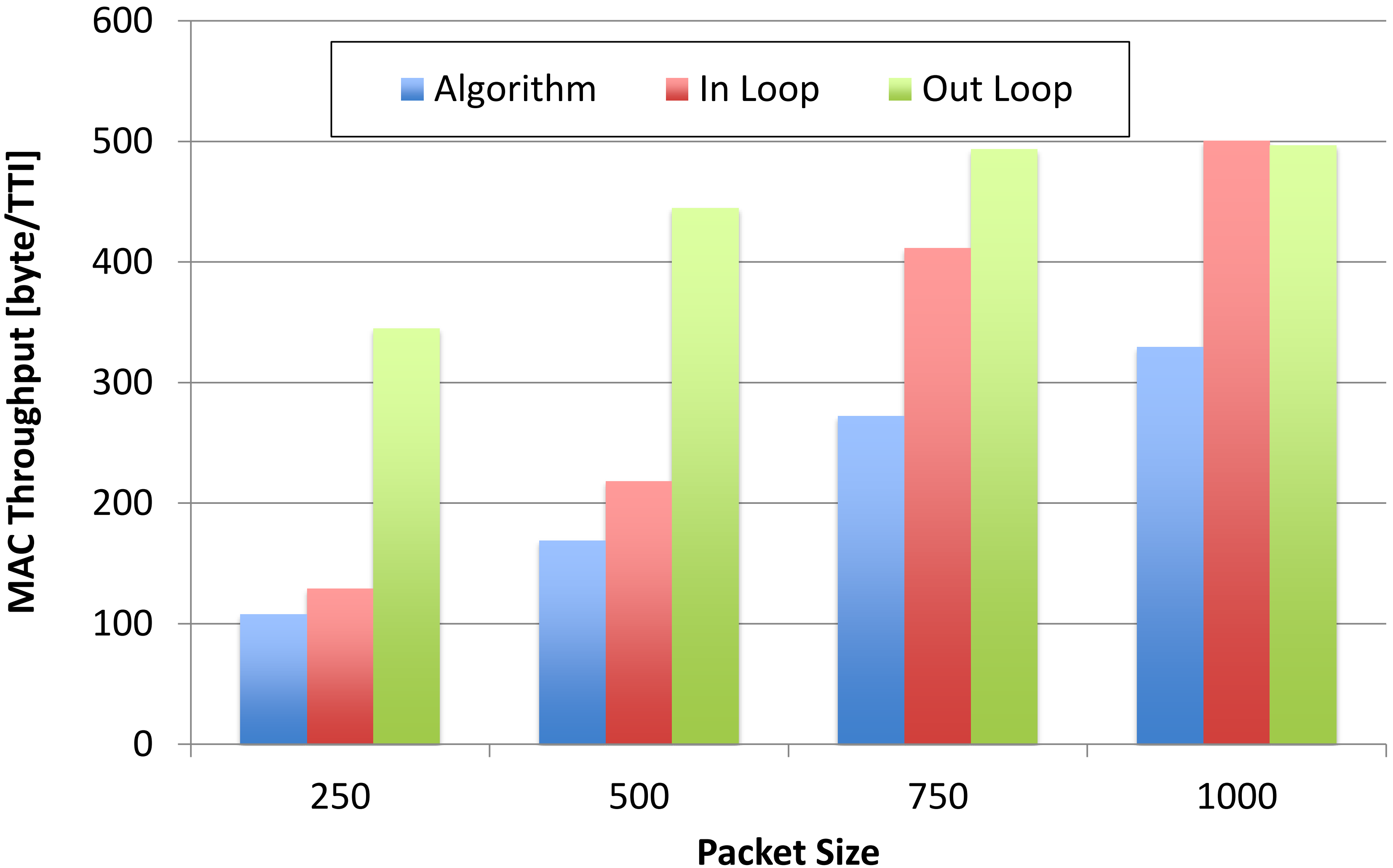


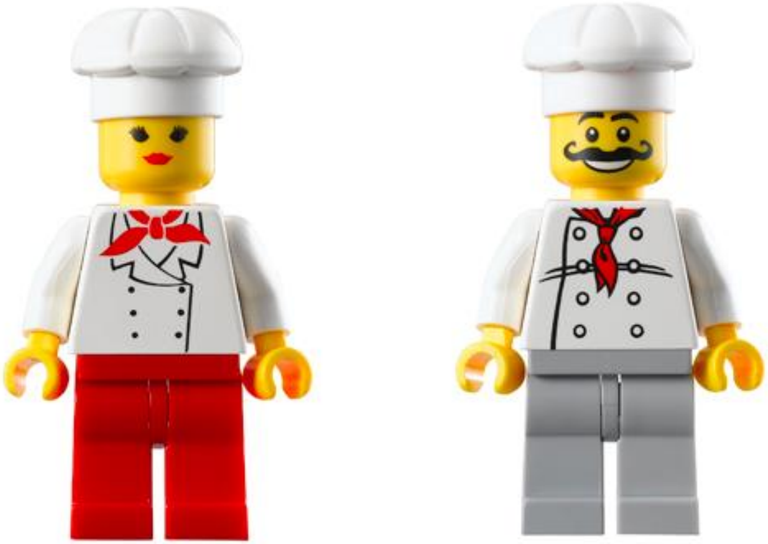
- Varying packet size
- Linear Mobility
- InLoop vs OutLoop

InLoop vs OutLoop



InLoop vs OutLoop





Any question while running for **dinner**?

Antonio Virdis

a.virdis@iet.unipi.it

simulte.com