# Quid Pro Quo: A Fair Linking Mechanism

Agustín Santos, Antonio Fernández Anta, Luis López
Institute Imdea networks and Universidad Rey Juan Carlos

Developing the Science of Networks

institute IMdea networks

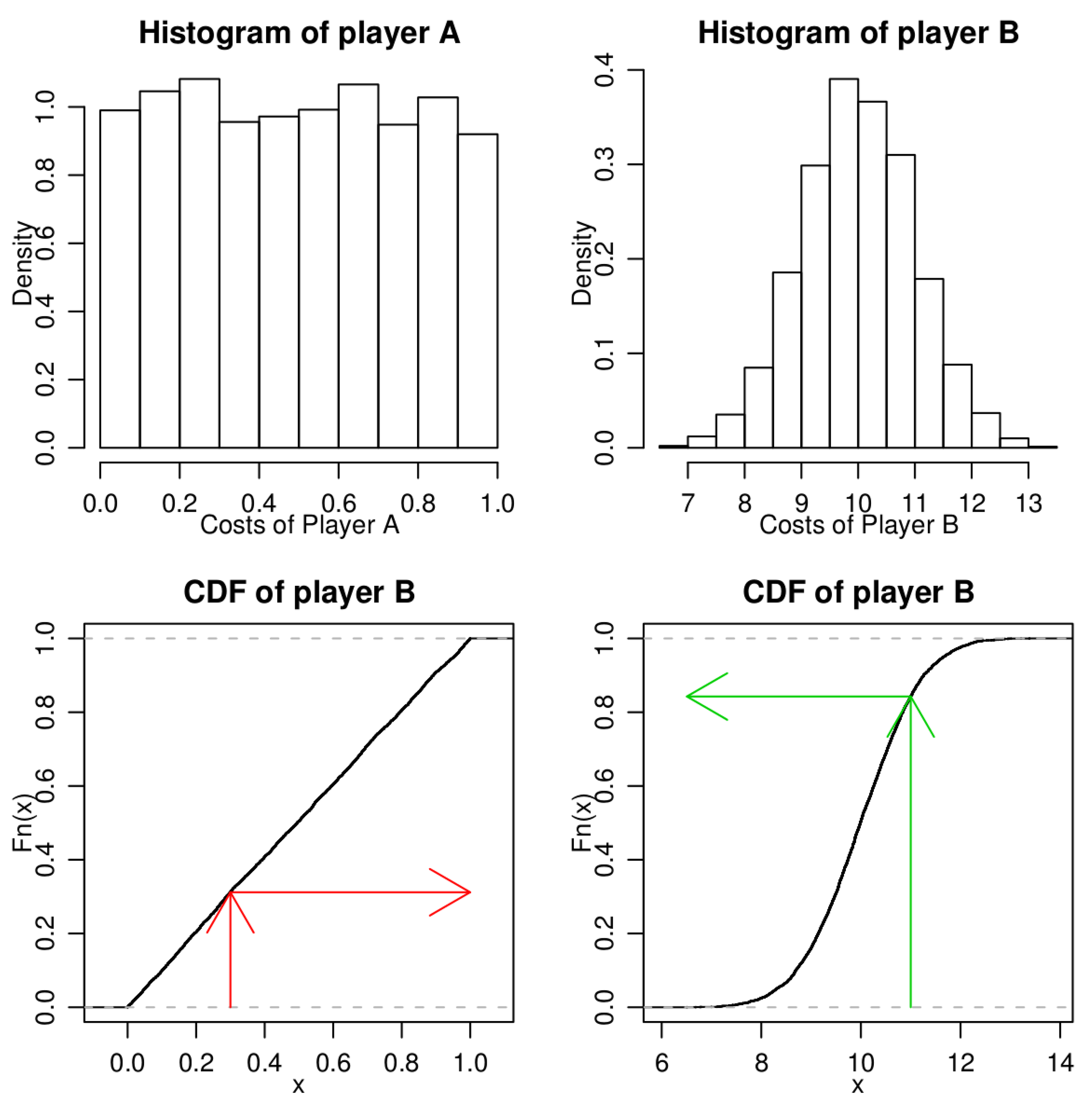
## Introduction

Resource allocation on Internet is one of the most relevant problems for computing systems. In such contexts, it is difficult to set up payments systems. An example of this could be task assignment on a P2P system.

Our work explores applications of the Linking Mechanism of Jackson and Sonnenschein[1] to Distributed Systems where player's distributions are unknown.

## Approach

Players may have heterogeneous preference's units. Or they could have several different distributions among them. When fairness is a requirement then we need to look for some normalization process.




Our proposal is based on the Probability Integral Transform (PIT). This transformation produces an uniform distribution for each player.

## Practical Implementation

A practical algorithm has been proposed and simulated, using the test of Kolmogorov-Smirnov as Good of Fit (GoF) test.

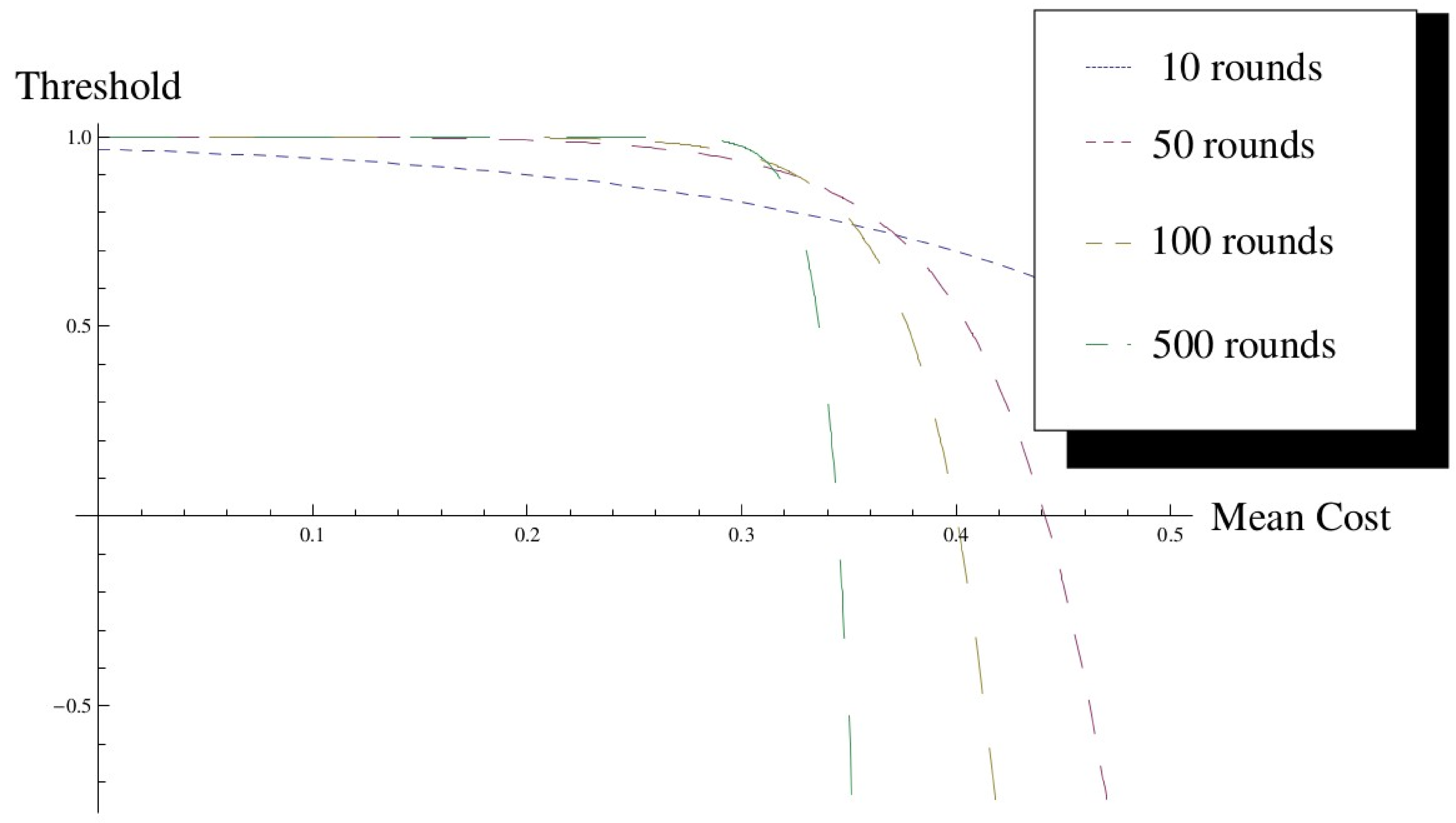
In practice, the statistic requires a relatively large number of data points to properly reject the null hypothesis. This is a problem for a practical implementation on distributed systems.

In our case, we use the actual and expected outcome in order check confidence limits.

## Adapting KS Confidence Limits

p-value threshold is adapted for each round (k) using the actual payoff ($\mu_k$) and the theoretical one ($\mu$).

$$p\text{-}th_k = \frac{1}{\log(k+1)^{\delta\cdot(1-(\mu_k-\mu)\cdot\sqrt{k})}},$$

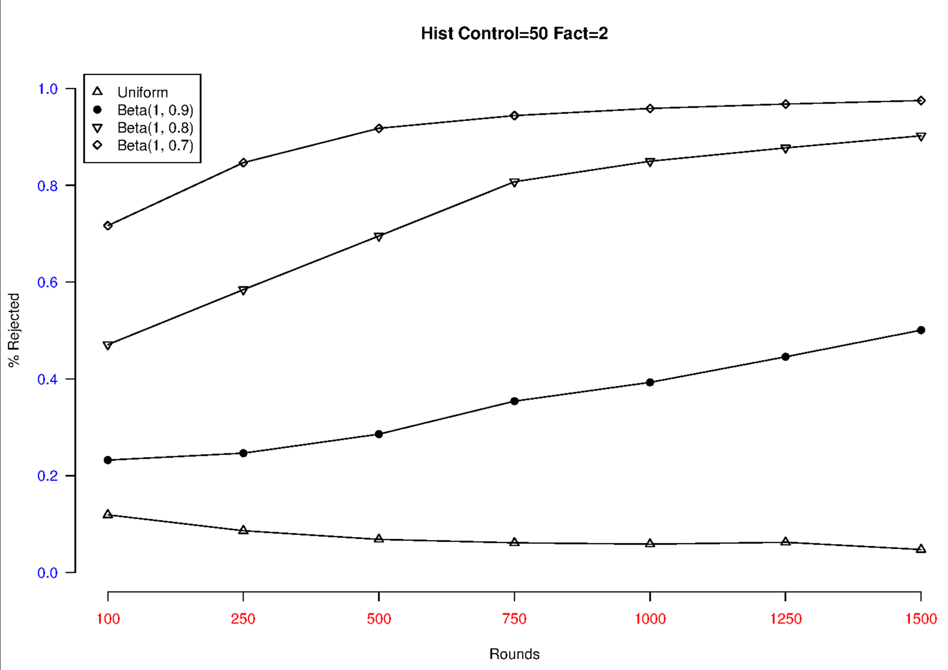



## Practical Implementation

**Algorithm 1** Implementable Quid Pro Quo mechanism (code for node $i$)

1: Estimate the cost $c_i$ of the assigment
2: Publish the normalized cost $\bar{c}_i = PIT(c_i)$
3: Wait to receive the normalized values $\bar{c}_j$ from the other players
4: **for all** $j \in N$ **do**
5:     Let $p\text{-}th_j \leftarrow \frac{1}{\log(k+1)^{\delta(1-(\mu_{j,k}-\mu)\cdot\sqrt{k})}}$
6:     **if** not $KSTest(\bar{c}_j, Historic_j, p\text{-}th_j)$ **then**
7:         $\bar{c}_j \leftarrow Random(\bar{c}_{-j})$
8:     **end if**
9:     $Historic_j \leftarrow Historic_j \cup \{\bar{c}_j\}$
10: **end for**
11: Let $d = \underset{j \in N}{argmin}\, \bar{c}_j$
12: **if** $d = i$ **then**
13:     resource (i.e task) is assigned to $i$
14: **else**
15:     do nothing (i.e. node $d$ will execute the task)
16: **end if**
17: **for all** $j \in N$ **do**
18:     recompute $\mu_{j,k+1}$
19: **end for**

## Simulations




By performing simulations, we have checked various aspects of our proposal.

The performance loss caused by false negatives is minimal .

We have simulated dishonest behavior by using several distributions close to the uniform. The dishonest get into trouble rapidly.

## Reference

[1] Jackson, M.O., Sonnenschein, H.F.: Overcoming incentive constraints by linking decisions. Econometrica 75(1) (2007)