Establishing the equivalence between operators

Gonzalo P. Rodrigo - gonzalo@cs.umu.se

January 28, 2014

1 Introduction

Decentralized prioritization is a technique to influence job scheduling order in grid fairshare scheduling without centralized control. The technique uses an algorithm that measures individual user distances between quota allocations and historical resource usage (intended and current system state) to establish a semantic for prioritization.

As described in [1], the intended target and current are represented in trees, a natural map of the organizations that represent. To calculate the distance between the target and the state, priority operators are applied to each corresponding to generate a priority tree. Which later will be manipulated to calculate the final priority.

The priority operates are mathematical functions defined by the following constraints (Extracted from [2]):

1. An operator is a function with two input variables such that:

$$t \in [0,1], s \in [0,1] \Rightarrow F(t,s) \in [-1,1] \tag{1}$$

$$\begin{aligned} F(t,s) = 0 &\iff t = s \\ F(t,s) > 0 &\iff t > s \\ F(t,s) < 0 &\iff t < s \end{aligned} \tag{2}$$

   where t represents a target value, s a (normalized) state value and $t = s$ is the ideal balance axis transecting the operator value space.

2. The function is strictly increasing on target and strictly decreasing on state:

$$\begin{aligned} &\forall t_j, t_i, s_j, s_i, t, s \in (0,1], \\ F(t_j,u) > F(t_i,u) &\iff \quad t_j > t_i \\ F(t_j,s_i) > F(t_j,s_j) &\iff \quad s_j < s_i \\ F(t_j,s) = F(t_i,s) &\iff \quad t_j = t_i \\ F(t,s_i) = F(t,s_j) &\iff \quad s_j = s_i \end{aligned} \tag{3}$$

3. Operator functions are idempotent and deterministic.

In this document study the operators in the environment of the job ordering, looking for a way to establish an *equivalence*: for any possible input paris $(t_j, s_j), (t_i, s_i)$ they would always other them in the same way.

In this document we will present a sufficient condition to establish that two operators are equivalent.

2 Equivalence Theorem

1. Two operators F, F' will be equivalent only if the produce the same ordering for any given input pairs $(t_j, s_j), (t_i, s_i)$:

$$\begin{aligned} &\forall t_j, s_j, t_i, s_i \in [0,1], \\ F(t_j, s_j) = F(t_i, s_i) &\iff F'(t_j, s_j) = F'(t_i, s_i) \wedge \\ F(t_j, s_j) < F(t_i, s_i) &\iff F'(t_j, s_j) < F'(t_i, s_i) \end{aligned} \tag{4}$$

2. To state that two operators F, F' are equivalent it is sufficient condition to prove that:

$$\begin{aligned} \forall t_j, s_j, t_i, s_i \in [0,1], \exists G(F, t_j, s_j, t_i) = s_i : \\ F(t_j, s_j) = F(t_i, s_i) \wedge G(F, t_j, s_j, t_i) = G(F', t_j, s_j, t_i) \end{aligned} \tag{5}$$

3. As a consequence of the equations 4 and 5 we can then state:

$$\begin{aligned} \forall t_j, s_j, t_i, s_i \in [0,1], \exists G(F, t_j, s_j, t_i) = s_i : \\ F(t_j, s_j) = F(t_i, s_i) \wedge G(F, t_j, s_j, t_i) = G(F', t_j, s_j, t_i) \Rightarrow \\ \begin{pmatrix} F(t_j, s_j) = F(t_i, s_i) \iff F'(t_j, s_j) = F'(t_i, s_i) & \wedge \\ F(t_j, s_j) > F(t_i, s_i) \iff F'(t_j, s_j) > F'(t_i, s_i) & \end{pmatrix} \end{aligned} \tag{6}$$

3 Demonstration base

3.1 Method

In this section we will follow the demonstration of the equation 6. The starting point are the following premises:

- Considered operators are monotonically increasing on p
- Considered operators are monotonically decreasing on u

Both premises come from the definition of the priority operator stated in the equations of 3.

At this point we continue with the theorem. In order to prove it, we have to prove 2 conditions:

1.

$$\begin{aligned} \forall t_j, s_j, t_i, s_i \in [0,1], \exists G(F, t_j, s_j, t_i) = s_i : \\ F(t_j, s_j) = F(t_i, s_i) \wedge G(F, t_j, s_j, t_i) = G(F', t_j, s_j, t_i) \Rightarrow \\ \big(F(t_j, s_j) = F(t_i, s_i) \iff F'(t_j, s_j) = F'(t_i, s_i)\big) \end{aligned} \tag{7}$$

2.

$$\begin{aligned} \forall t_j, s_j, t_i, s_i \in [0,1], \exists G(F, t_j, s_j, t_i) = s_i : \\ F(t_j, s_j) = F(t_i, s_i) \wedge G(F, t_j, s_j, t_i) = G(F', t_j, s_j, t_i) \Rightarrow \\ \big(F(t_j, s_j) > F(t_i, s_i) \iff F'(t_j, s_j) > F'(t_i, s_i)\big) \end{aligned} \tag{8}$$

In the next section we will develop one by one.

3.2 Development

3.2.1 Equal statement

Statement to prove:

$$\begin{aligned} \forall t_j, s_j, t_i, s_i \in [0,1], \exists G(F, t_j, s_j, t_i) = s_i : \\ F(t_j, s_j) = F(t_i, s_i) \wedge G(F, t_j, s_j, t_i) = G(F', t_j, s_j, t_i) \Rightarrow \\ \big(F(t_j, s_j) = F(t_i, s_i) \iff F'(t_j, s_j) = F'(t_i, s_i)\big) \end{aligned} \tag{9}$$

By the definition of G:

$$\begin{aligned} G(F, t_j, s_j, t_i) = s_i = G(F', t_j, s_j, t_i) \wedge F(t_j, s_j) = F(t_i, s_i) \quad \Rightarrow \\ F'(t_j, s_j) = F'(t_i, s_i) \end{aligned} \tag{10}$$

Proving one direction of the double implication. As F and F' can be exchanged, the other implication is also proved automatically.

3.2.2 Greater-than statement

Statement to proof:

$$\begin{aligned} \forall t_j, s_j, t_i, s_i \in [0,1], \exists G(F, t_j, s_j, t_i) = s_i : \\ F(t_j, s_j) = F(t_i, s_i) \wedge G(F, t_j, s_j, t_i) = G(F', t_j, s_j, t_i) \Rightarrow \\ \big(F(t_j, s_j) > F(t_i, s_i) \iff F'(t_j, s_j) > F'(t_i, s_i)\big) \end{aligned} \tag{11}$$

In the equation 10 we proved that $F(t_j, s_j) = F(t_i, s_i) \iff F'(t_j, s_j) = F'(t_i, s_i)$. From the definition of the priority operators in the equations 4, we know that F and F' are constantly increasing on t and decreasing on s, by mixing this two statements:

$$\begin{aligned} F(t_j, s_j) > F(t_i, s_i) \wedge \exists t_k > t_i : G(F, t_j, s_j, t_k) = s_i \wedge G(F', t_j, s_j, t_k) = s_i \Rightarrow \\ t_k > t_i \wedge F'(t_j, s_j) = F'(t_k, s_i) \Rightarrow F'(t_j, s_j) > F'(t_i, s_i) \end{aligned} \tag{12}$$

Proving the first part of the statement. As F and F' can be exchanged, the double implication is probed.

References

[1] Östberg, P-O. and Espling, D., Elmroth, E.: Decentralized scalable fairshare scheduling. Future Generation Computer Systems - The International Journal of Grid Computing and eScience 29 (2013) 130–143

[2] Rodrigo, G.P., Östberg, P-O.and Elmroth, E.: Priority operators for fairshare scheduling. Not yet publisehd (2014)