Reducing accumulated relative tardiness (ART) for soft realtime load balancing

Abstract

Many systems today have incorporated in them the idea of task priorities and have a growing requirement of soft real-time guarantees while making scheduling (load balancing decisions). In this project, I am looking at the idea of using reinforcement learning to tune weights of the weighted round robin (WRR) load balancer to distribute resource between tasks to maximize the overall tardiness in a system.

Introduction

I am looking at load balancing, possibly in edge clusters, as soft real time system. This system might have deadlines for different processess or for processes generated by certain sources. But may not have any deadline requirements for many others.

- Look at the processing of various processes of the system and form different clusters to form a loose definition of different states of the system.
  - Advantage is that this can allow us define a finite number of states for any system.
  - A first attempt can select number of clusters based on number of different requests that can be made to the system. However, higher dimensional cluster like a combination of request types can also be considered.
- Once a determination of states are made, use reinforcement learning to find an optimal scheduling solution that looks to reduce tardiness given a generic optimization goal.
  - Optimization goal can be provided in terms of a general service time.
  - ***tardiness*** $:= max[0, (t - d)]$
    1. t -> total time spent in the system by a process
    2. d -> deadline for the process. If the process does not have a specific deadline then d defaults to the optimization goal.

Project Goal

$$Goal := min \sum_{i=1}^{N(T)} max[0, \frac{(t_{ij} - d_{ij})}{d_{ij}}]$$

So,

$$G_t = max \sum_{i=1}^{N(T)} max[0, -\frac{(t_{ij} - d_{ij})}{d_{ij}}]$$

where,

$$d_{ij} = \mu_{t_{ij}} + c_j \sigma_{t_{ij}}$$

>

$$c_j = 1 + \frac{1}{p_j}$$

> T is the ***time interval at which the reward is measured*** and $N(T)$ is the ***number of jobs that go through that system in that time***.

where $p_{ij} :=$ ***priority of task type j*** and $c_j \in (1, 2)$

$\frac{(t_{ij} - d_{ij})}{d_{ij}}$ is the ***relative tardiness*** of the job. We are interested in minimizing the ***accumulated relative tardiness (ART)*** - $\sum_{i=1}^{N(T)} max[0, \frac{(t_{ij} - d_{ij})}{d_{ij}}]$ of the overall system.

States and storing them

The system will be under an initial period of observation while it's under scripted load. These observations are to be used to create a broad definition of "states" which will consist of a bunch of similar actual states of the system. We will add the actual parameters later.

There can be a predefined number of distinct states K, possibly depending on the dimensionality of the data being considered. Each state can be stored as a mean vector $\bar{\mu}$ and a covariance matrix Σ. The states can be recalibrated if the number of observed data points that lie outside all states; $\bar{s} \notin \{\bar{\mu_k} \pm 3\Sigma_k\} \forall k \in K$; cross a predefined threshold, $N(k)$. Recalibration will include another observation stage.

Load Balancing

The Weighted Round Robin (WRR) forms the basis of the load balancing algorithm. Different types of tasks have different priorities. For each task type, the backend servers can have different weights. The idea is to readjust the weights of the servers, for each task type, so that the overall resources are distributed between the jobs in the system so as to maximize G_t.

Training the RL model

TODO