University of Duisburg-Essen, Institute for Experimental Mathematics

Member Selection Policies for the Reliable Server Pooling Protocol Suite

Thomas Dreibholz

Institute for Experimental Mathematics

University of Duisburg-Essen, Germany

dreibh@exp-math.uni-essen.de

http://www.exp-math.uni-essen.de/~dreibh

Table of Contents

Thomas Dreibholz's Reliable Server Pooling Page
http://tdrwww.exp-math.uni-essen.de/dreibholz/rserpool/

Introduction

- draft-ietf-rserpool-common-param-06.txt and draft-ietf-rserpool-asap-09.txt:
  - Incomplete definition of 4 policies

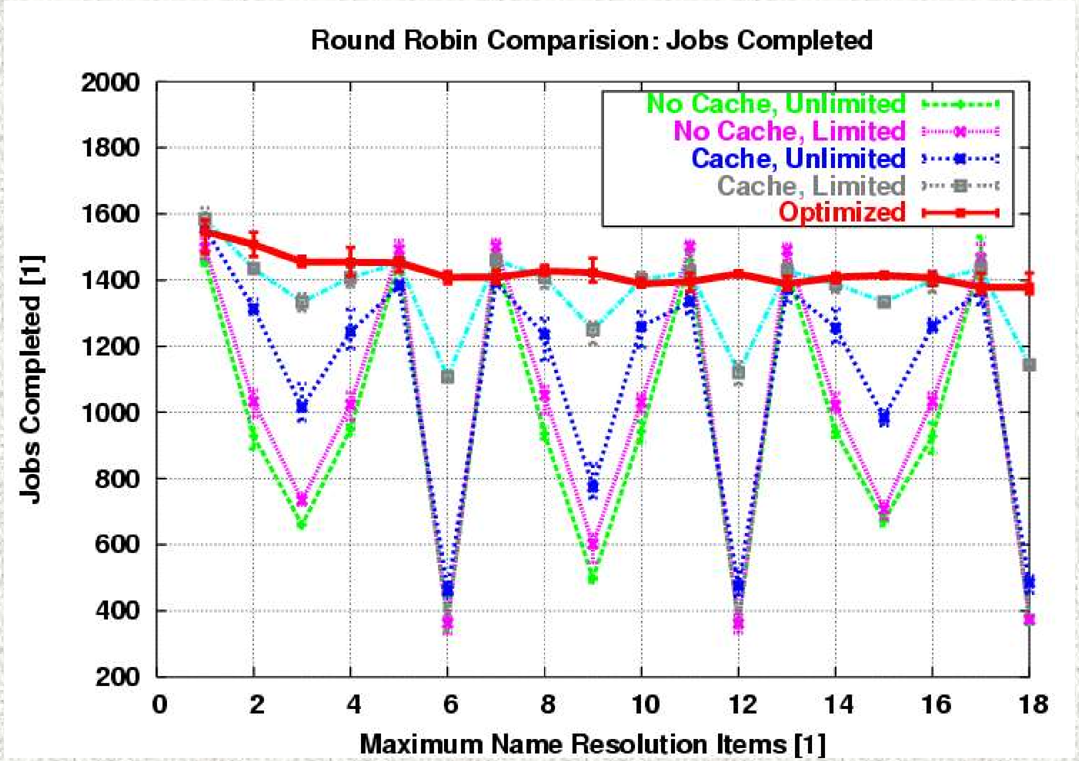
- Our Policy Examinations
  - Completing the definition of the 4 policies from the draft
  - Simulation prototype for examining pool policies
  - Adding new useful policies
  - => draft-tuexen-rserpool-policies-00.txt

- Our Recommendation:
  - Definition of only one default policy (Round Robin) in ASAP document
  - Definition of other (optional) policies as separate document

Round Robin Problem

- **Problem Example:**
  - NS selects 3 PEs as result of an ASAP Name Resolution in Round Robin order, i.e. the first 3 ones, then the next 3 ones and so on ...
  - PU selects **one** of the 3 in Round Robin order, i.e. always the first one
  - PU uses the selected PE's service
  - Result:
    - Worst case: Pool size is divisible by 3
    - PE 1, 4, 7, 10, ..., 1+(3*k) are used
    - All other PEs will remain idle
- **Our Proposed Solution:**
  - NS steps forward by **only one PE** or
  - Maximum amount of PEs to be returned as Name Resolution Response is **randomized**

Round Robin Simulation Results



- 18 PEs, provide computation service
- 36 PUs, request jobs to be computed
- 3 NS

- **Goal**: Pool should process as many jobs as possible

Cache:	PU Name Resolution cache (stale value: 30s)
No Cache:	No PU Name Resolution cache
Limited:	PE accepts limited amount of simultaneous requests, rejects additional ones
Unlimited:	PE accepts unlimited amount of simultaneous requests
Optimized:	Randomizing amount of PEs to be returned from 1 to 17

Least Used with Degradation

- draft-ietf-rserpool-common-param-06.txt:

  „This policy is the same as the Least Used policy with the exception that, each time the PE with the lowest policy value is selected from the Pool as the receiver of the current message, its policy value is *incremented*, and thus it may no longer be the lowest value in the Pool.“

- Question: Incremented by what? Value depends on:
  – PE's request handling power and probability, that PU really uses the selected PE

- Proposed Solution:
  – Add a load degradation constant to the policy
  – Each PE may use its own constant (faster PE -> lower value)

- Examples:
  1. Load := Current CPU load in percent -> Load Degradation := 10%

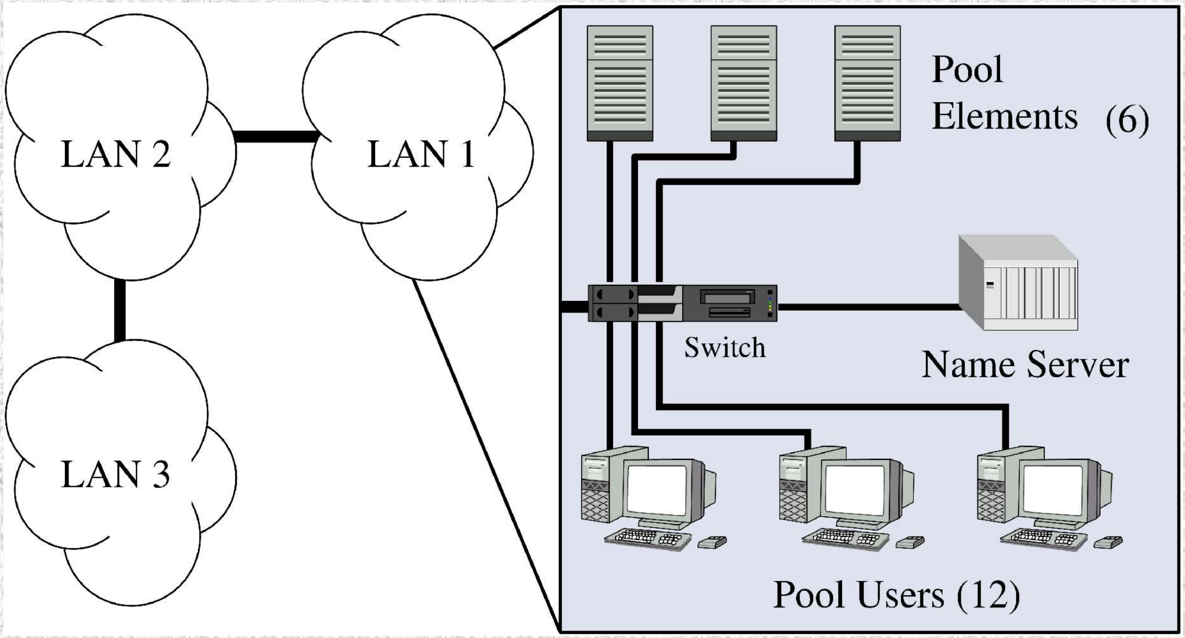
     (a fast PE could set its Load Degradation to e.g. 5%)
  2. Load := Current Amount of users -> Load Degradation := 1

New Policies

- Random/Weighted Random
  - Non-deterministic selection
- Priority Least Used – an improvement of Least Used
  - Least Used:
    - Get PE that has lowest load
    - Round robin (or random) selection between PEs of equal load
  - Problem Example:
    - PE #1: Comp. Power: 10 Requests/s, 50% Loaded
    - PE #2: Comp. Power: 100 Requests/s, 55% Loaded
    - => Least Used will select PE #1, although PE #2 would handle a new request faster
  - Solution:
    - Add per-PE load increment constant
    - Select PE where *Current Load* + *Load Increment* is lowest

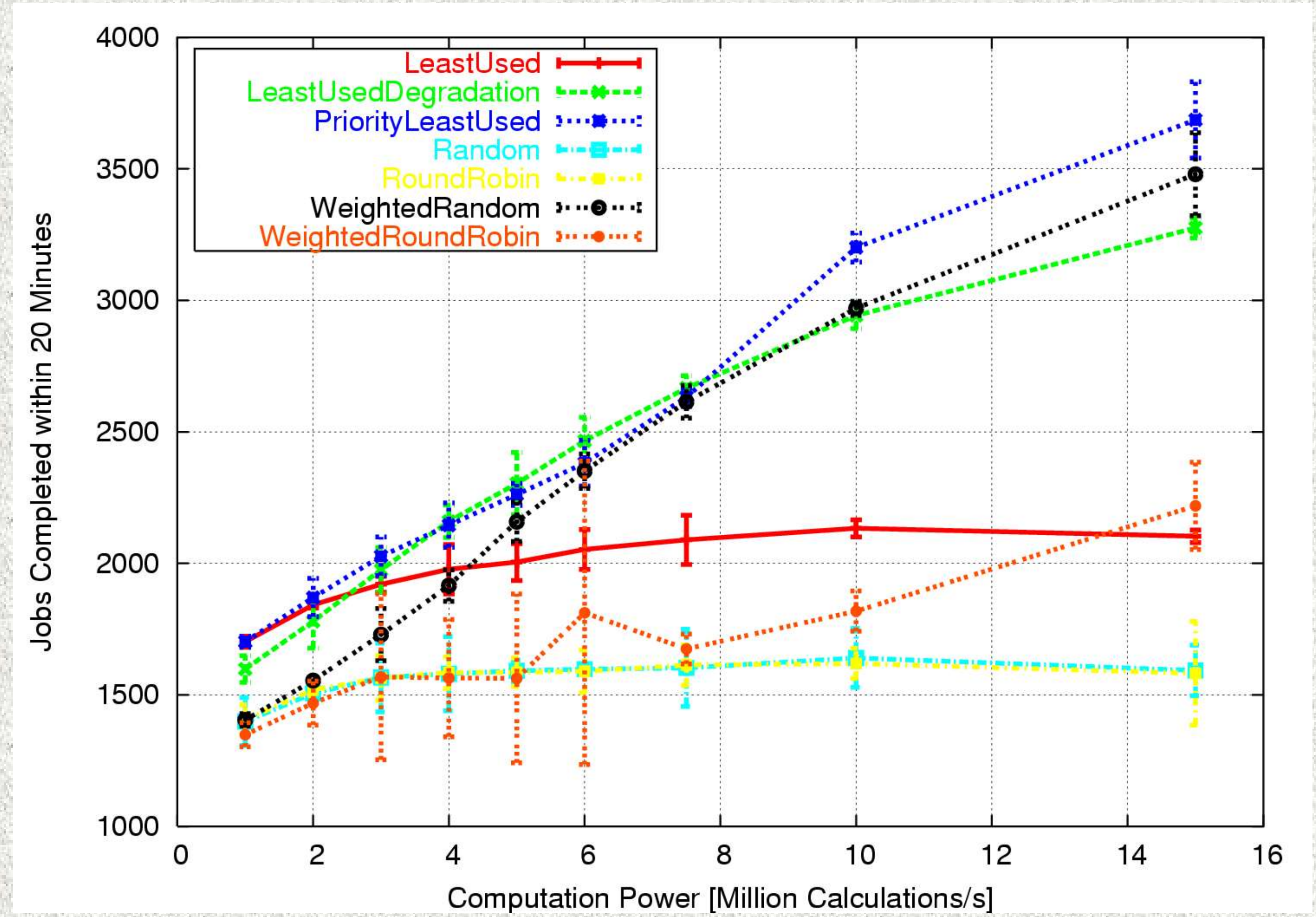
Simulation Results (I)

- Scenario:
  - 3 LANs
    - 1 NS
    - 6 PEs
    - 12 PUs
  - PEs:
    - Comp. Power:
      1M calcs/s
  - PUs:
    - Request computation from pool
    - Average request size: 10M calculations, exponentially distributed
    - Average inter-request time: 10s, exponentially distributed



- Increase computing power of each LAN's first PE (no change for other PEs)
- **Goal**: Pool should be able to utilize the higher computation power

Simulation Results (II)

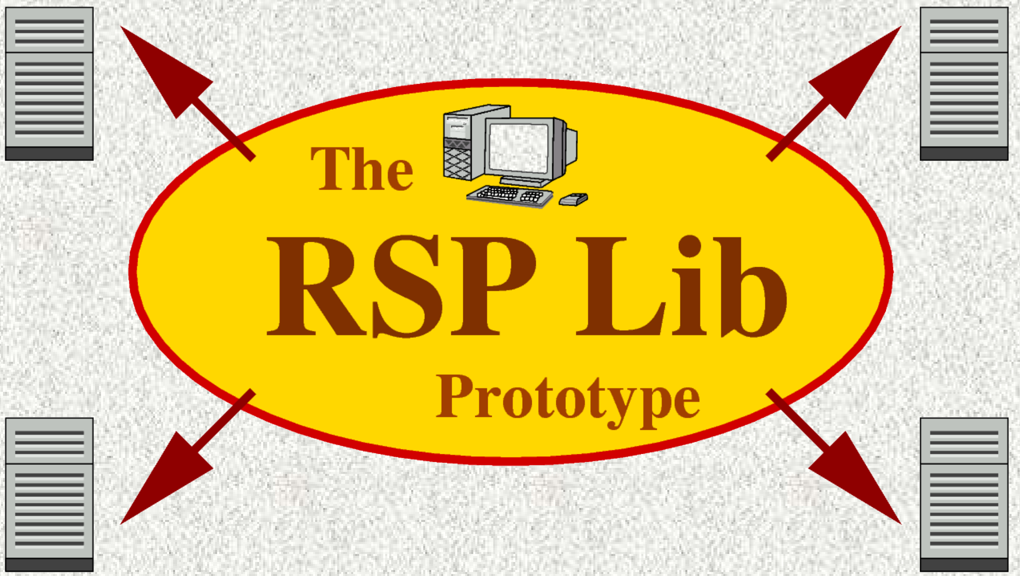


Conclusions

- We completely defined the four policies of
  - draft-ietf-rserpool-common-param-06.txt
  - draft-ietf-rserpool-asap-09.txt
- We defined new policies, performing better in certain scenarios
- New policies will be defined in the future
  - For specific applications
  - For specific scenarios

- Our proposal:
  - Define one mandatory default policy in ASAP draft: Round Robin
  - Create new document for optional policies
  - => draft-tuexen-rserpool-policies-00.txt

Any Questions?



Project Homepage:

http://tdrwww.exp-math.uni-essen.de/dreibholz/rserpool/

Thomas Dreibholz, dreibh@exp-math.uni-essen.de