Load Balancing Proposal for Reducing Runtime in Multiprocessed Environments

Vinicius Ribas Samuel dos Santos, Ana Karina Machado, Edson Luis Padoin

Presentation Script

1. Introduction
2. Goal
3. SmartLB
4. Methodology
5. Results
6. Conclusion
7. Future Works

Summary

1. **Introduction**
2. Goal
3. SmartLB
4. Methodology
5. Results
6. Conclusion
7. Future Works

1. Introduction

- One of the hardware components that presented a huge evolution was the processor;
- Most of the time there is a concern with the load unbalance generated by certain applications;
- Faced with this problem is that load balancers are developed aiming at an equilibrium of loads on the processors.

Summary

1. Introduction
2. **Goal**
3. SmartLB
4. Methodology
5. Results
6. Conclusion
7. Future Works

2. Goal

- This paper presents a proposal for a new load balancer for reducing the execution time of parallel applications executed in multiprocessed environments.

Summary

1. Introduction
2. Goal
3. **SmartLB**
4. Methodology
5. Results
6. Conclusion
7. Future Works

3. SmartLB

- Was created over improvements in the strategies used in GreedyLB and RefineLB algorithms;
- Balance the loads between processors reducing the number of migrations;
- The implementation uses a centralized approach;
- It adopts a threshold to define the acceptable load unbalance, reducing the number of migrations.

3. SmartLB

Implementation of the SmartLB algorithm:

```
1   PM = getCargaMaior();
2   Pm = getCargaMenor();
3   if((Pm/PM) > Threshold) {
4       for(i = 1; i <= nTarefas; i++) {
5           if(getProcessadorAtual(i) == PM) {
6               ct = getCargaTarefa(i);
7               D = PM - Pm;
8               if(ct <= D) {
9                   migrarProcesso(i, PM, Pm);
10                  PM = getCargaMaior();
11                  Pm = getCargaMenor();
12              }
13          }
14      }
15  }
```

Summary

1. Introduction
2. Goal
3. SmartLB
4. **Methodology**
5. Results
6. Conclusion
7. Future Works

4. Methodology

- The benchmark used to perform the tests was the lb_test;
- The plataform used to execute the tests:
  - **Processor:** Intel i7-6500U.
  - **Operacional System:** Linux Ubuntu 16.04
  - **Kernel:** 4.4.33-1.
  - **Charm ++:** 6.5.1.
  - **g++:** 5.4.1.

4. Methodology

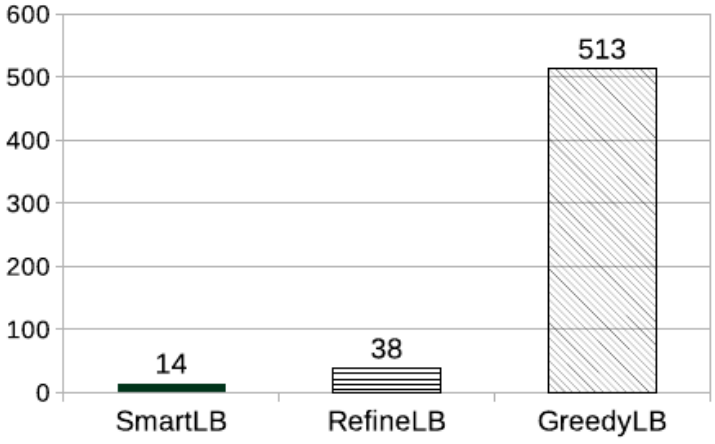
- To perform the tests, the three balancers were applied to a simulation using 150 iterations per task;
- The load balancer synchronizations were defined every 10 iterations;
- The number of chares varied from 50 and 100 tasks;
- With a computational load that can vary between 1500 and 150000;
- During the tests, we took into account the total execution time and the total of objects migrated between the processors.

Summary

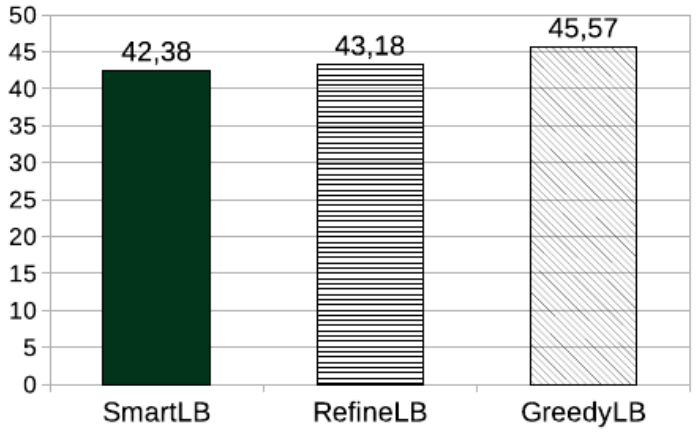
1. Introduction
2. Goal
3. SmartLB
4. Methodology
5. **Results**
6. Conclusion
7. Future Works

5. Results

- Test results using benchmark lb_test with 50 tasks:



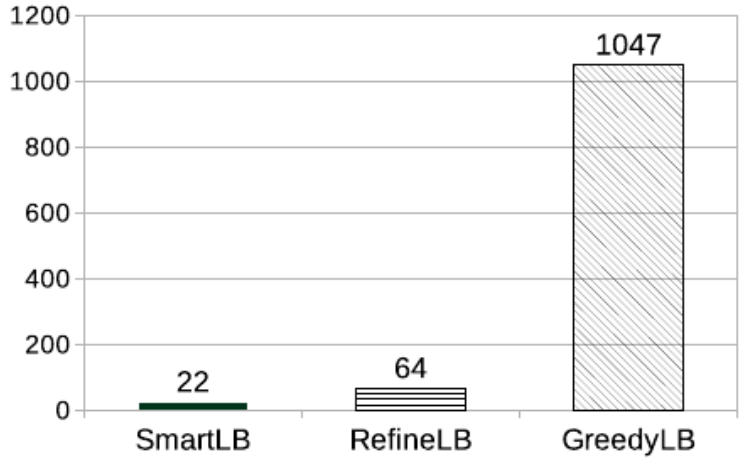
Total migrated tasks



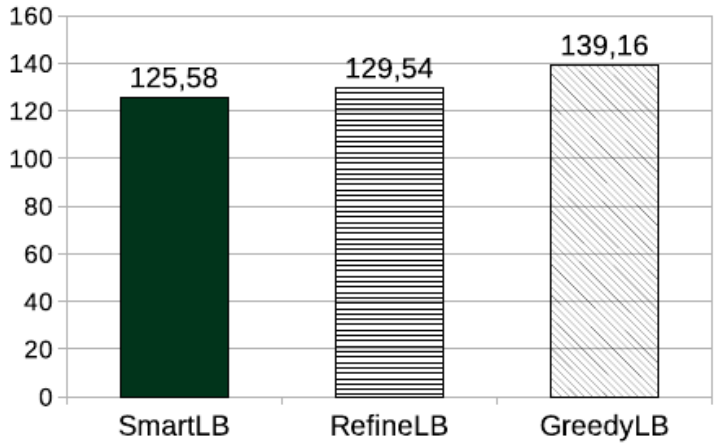
Total execution time (s)

5. Results

- Test results using benchmark lb_test with 100 tasks:



Total migrate tasks



Total execution time (s)

Summary

1. Introduction
2. Goal
3. SmartLB
4. Methodology
5. Results
6. **Conclusion**
7. Future Works

6. Conclusion

- Comparing the results, we conclude that SmartLB showed a better performance in object migrations;
- The Load Balancer is performing a good control of migrated objects, avoiding waste and collaborating for agile and precise load balancing.

Summary

1. Introduction
2. Goal
3. SmartLB
4. Methodology
5. Results
6. Conclusion
7. **Future Works**

7. Future Works

- To realize improvements in the algorithm SmartLB aiming to manage even more the control of migrations of tasks;
- Perform tests on large parallel machines in order to solve real problems.

Load Balancing Proposal for Reducing Runtime in Multiprocessed Environments

Vinicius Ribas Samuel dos Santos, Ana Karina Machado, Edson Luis Padoin

Thanks!

References

- Arruda, G., Padoin, E. L., Pilla, L. L., Navaux, P. O. A., and Mehaut, J.-F. (2015). Proposta de balanceamento de carga para redução de migração de processos em ambientes multiprogramados. In XVI Simpósio de Sistemas Computacionais (WSCAD-WIC), pages 1–8, Florianópolis, RJ;
- Bang-Jensen, J., Gutin, G., and Yeo, A. (2004). When the greedy algorithm fails. Discrete Optimization, 1(2):121–127;
- Freytag, G., Arruda, G., Martins, R. S. M., and Padoin, E. L. (2015). Análise de desempenho da paralelização do problema de caixeiro viajante. In XV Escola Regional de Alto Desempenho (ERAD), pages 1–4, Gramado, RS. SBC;

References

- Jyothi, R., Lawlor, O. S., and Kale, L. V. (2004). Debugging support for charm++. in parallel and distributed processing symposium. page 264;
- Padoin, E., Castro, M., Pilla, L., Navaux, P., and Mehaut, J.-F. (2014). Saving energy by exploiting residual imbalances on iterative applications. In High Performance Computing (HiPC), 2014 21st International Conference on, pages 1–10;
- Pilla, L. L. and Meneses, E. (2015). Programação paralela em charm++. pages 1–20.