Load balancing by an asynchronous greedy algorithm

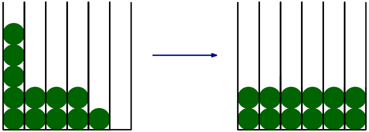
Abbas Mehrabian

Simons Institute

ITCS Graduating Bits, 9 January 2017

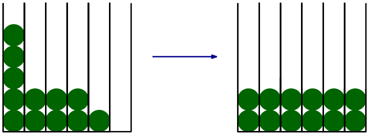
joint work with Petra Berenbrink, Peter Kling, Chris Liaw

Load balancing



Want to re-allocate balls into bins to achieve perfect balance quickly.

Load balancing



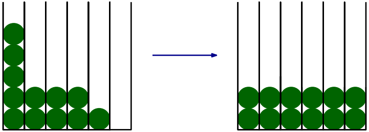
Want to re-allocate balls into bins to achieve perfect balance quickly.

Definition (Asynchronous greedy algorithm)

1. Each ball has an exponential clock of rate 1. When the clock rings, the ball is activated.
2. On activation, the ball chooses a random bin and moves there if its own load is improved by doing so.

Simple, distributed, asynchronous, ball-controlled, no global knowledge

Load balancing



Want to re-allocate balls into bins to achieve perfect balance quickly.

Definition (Asynchronous greedy algorithm)

1. Each ball has an exponential clock of rate 1. When the clock rings, the ball is activated.
2. On activation, the ball chooses a random bin and moves there if its own load is improved by doing so.

Simple, distributed, asynchronous, ball-controlled, no global knowledge

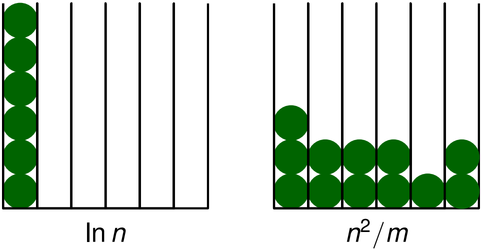
n = number of bins, m = number of balls

$O(n^2)$ Bound on expected time to reach perfect balance [Goldberg'04]

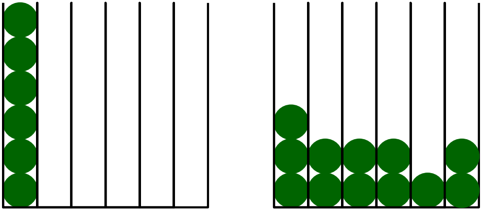
$O\left(\ln(n)^2 + \ln(n) \cdot n^2/m\right)$ [Ganesh,Lilienthal,Manjunath,Proutiere,Simatos'12]

$O(\ln n + n^2/m)$ [Berenbrink, Kling, Liaw, M'17]

Tightness of our analysis: $O(\ln n + n^2/m)$



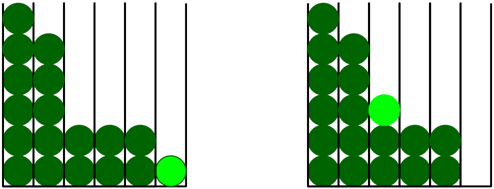
Tightness of our analysis: $O(\ln n + n^2/m)$



This algorithm is known as randomized local search.

We also show, whp, time to reach perfect balance $\leq O(\ln n + \ln n \cdot n^2/m)$

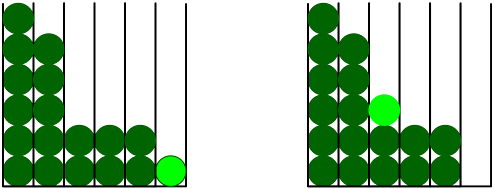
About the proof



A key majorization lemma:

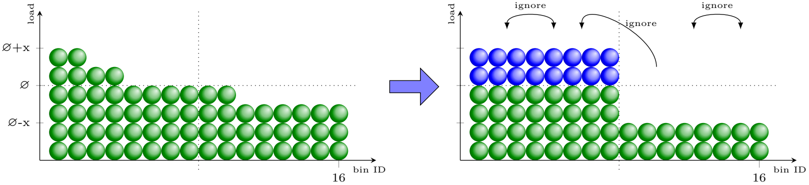
Balancing time of left configuration $\preccurlyeq$ Balancing time of right configuration

About the proof

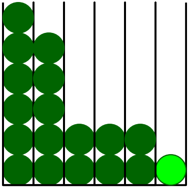


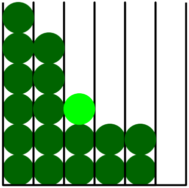
A key majorization lemma:

Balancing time of left configuration $\preccurlyeq$ Balancing time of right configuration
Helps in two ways: (1) we may do some destructive moves to make "well-shaped" configurations that are simpler to analyze.



About the proof



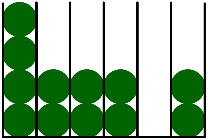


A key majorization lemma:

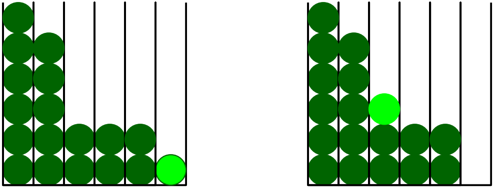
Balancing time of left configuration $\preccurlyeq$ Balancing time of right configuration

Helps in two ways: (1) we may do some destructive moves to make "well-shaped" configurations that are simpler to analyse.

(2) we may "ignore" certain (at the moment unwanted) moves made by the algorithm.



About the proof



A key majorization lemma:

Balancing time of left configuration $\preccurlyeq$ Balancing time of right configuration
Helps in two ways: (1) we may do some destructive moves to make "well-shaped" configurations that are simpler to analyse.
(2) we may "ignore" certain (at the moment unwanted) moves made by the algorithm.

1. max load $-$ min load is reduced to m/n within time $\leq O(\ln n)$
2. max load $-$ min load is reduced to $O(\ln n)$ within time $\leq O(\ln n)$
3. max load $-$ min load is reduced to 0 within time $\leq O(n^2/m)$

About me

Interests

- ✓ Stochastic processes with applications in TCS
- ✓ Theoretical machine learning

Homes

- ✓ 2015: graduated from U of Waterloo
  Joseph Cheriyan and Nick Wormald
- ✓ 2016: postdoc at U of British Columbia and Simon Fraser
  Petra Berenbrink and Nick Harvey
- ✓ 2017 (Spring): Simons Institute
  pseudorandomness and machine learning

About me

Interests

- ✓ Stochastic processes with applications in TCS
- ✓ Theoretical machine learning

Homes

- ✓ 2015: graduated from U of Waterloo
  Joseph Cheriyan and Nick Wormald
- ✓ 2016: postdoc at U of British Columbia and Simon Fraser
  Petra Berenbrink and Nick Harvey
- ✓ 2017 (Spring): Simons Institute
  pseudorandomness and machine learning
- ✓ Next home? Who knows?