Alcom-FT Technical Report Series
ALCOMFT-TR-03-107

Bounds for the Convergence Rate of Randomized Local Search in a Multiplayer, Uniform Resource Sharing Game *

Paul Goldberg †
Department of Computer Science
University of Warwick

November 28, 2003

Abstract

This paper studies a resource sharing game introduced by Koutsoupias and Papadimitriou, that is intended to model a set of users who share several internet-based resources. Some of the recent work on this topic has considered the problem of constructing *Nash equilibria*, which are choices of actions where each user has optimal utility given the actions of the other users. A related (harder) problem is to find sequences of utility-improving moves that lead to a Nash equilibrium, starting from some given assignment of resources to users.

We consider the special case where all resources are the same as each other. It is known already that there exist efficient algorithms for finding Nash equilibria; our contribution here is to show furthermore that Nash equilibria for this type of game are reached rapidly by *Randomized Local Search*, a simple generic method for local optimization. Our motivation for studying Randomized Local Search is that (as we show) it can be realised by a simple distributed network of users that act selfishly, have no central control and only interact via the effect they have on the cost functions of resources.

1 Introduction

Suppose we have a set of resources and a set of agents, and each agent has a task with a given positive weight. If each agent selects a single resource, then we assume each agent experiences a cost that is proportional to the sum of the weights of tasks of agents sharing that resource. If agents are able to change their selection, then a particular set of selections is in a sense stable provided that there is no possible change of resource by any agent that will lead to that agent's cost decreasing. In game-theoretic terms, this is a *pure Nash equilibrium*.

Recent work has considered the problem of finding Nash equilibria in this particular situation (see Section 1.3 for a summary). Specific problems include finding any Nash equilibrium for a given set of tasks and resources, and finding a short sequence of moves from a given set of resource selections to a Nash equilibrium. Given a starting set of selections, one would like to find a sequence of moves each of which is "self-improving" in the sense that when an agent moves from one resource to another, its cost should go down. It is known from [9, 7] that the process of repeatedly making self-improving moves must terminate at a Nash equilibrium, but little is know about how long this sequence must be.

In this paper we consider an algorithm that can be realised as a distributed scheme in which agents, at random points in time, make self-improving moves independently of each other. We consider the following simple algorithm. Initially each agent is assigned a resource, and then at each step, an agent and a resource are selected uniformly at random, and the agent moves to that resource if its resulting cost is lower. We call this *randomized local search* since it is essentially the same as the "randomized local search" studied in other contexts (see Section 1.3).

*This work was partially supported by the IST Program of the EU under contract number IST-1999-14186 (ALCOM-FT)
†http://www.dcs.warwick.ac.uk/∼pwg/, Department of Computer Science, University of Warwick, Coventry, CV4 7AL, United Kingdom.

We continue by giving some definitions and notation that are used subsequently, then give a summary of our results, followed by more details on background and related work. Section 1.4 shows how Randomized Local Search can be realised by a simple distributed scheme with no central control — we argue that it is important that an algorithm should have that property, in view of studies of "coordination ratio" for this particular situation (for example [3, 4, 5, 8, 9, 10, 12, 13, 15]), which is typically discussed as being a price that is paid for not having central control over the agents. The convergence rate bounds are presented in Section 2.

1.1 Problem Statement and Definitions

For positive integer z let $[z]$ denote the set $\{1, 2, \ldots, z\}$. An instance of the game consists of a set R of m resources, where resource ℓ, $\ell \in [m]$, has an associated *capacity* c_ℓ. In the case of uniform resources studied here, $c_\ell = 1$ for all ℓ. There is also a set T of n tasks (each task belonging to a unique agent), where for each $j \in [n]$, task j has an associated *weight* w_j. Each task has m actions available to it. The cost experienced by task j selecting resource ℓ is the sum of weights of tasks that select ℓ, divided by ℓ's capacity c_ℓ.

An *assignment* A is any single set of choices of resources made by tasks; formally $A \in [m]^n$. (An assignment is sometimes called a "pure profile" in the literature.)

Let $M(j, A)$ denote the resource that task j selects in assignment A.

Let $L(\ell, A) = \sum_{j \in [n]} \{w_j \ : \ M(j, A) = \ell\}$, i.e. the sum of weights of tasks that use resource ℓ.

Let $C(\ell, A) = c_\ell^{-1} L(\ell, A)$ be the cost to any task that uses resource ℓ in A. (For uniform resources, $C(\ell, A) = L(\ell, A)$.)

A *move* is the operation of transferring a task from a resource to a new resource that reduces the cost for that task. Following [8] we say that a task is unsatisfied if there exists a move it can make that results in a reduction of its cost. An *attempt* is the operation of selecting a task j uniformly at random and a resource ℓ uniformly at random, and transferring j to ℓ provided that the operation is a (self-improving) move. If the operation would not reduce the cost to j, we leave the assignment unchanged. An attempt is *successful* if it does actually result in a move.

Given these definitions, *Randomized Local Search* (RLS) consists quite simply of a sequence of attempts, and the general question is, what is the expected number of attempts required for convergence to Nash equilibrium.

1.2 Summary of Results

It is shown in [9, 7] that any sequence of selfish moves must converge, for the game under consideration. It is also known from [8] how to construct a sequence of selfish moves from a given assignment to a Nash equilibrium, in the case of uniform resources under consideration here. The following results address the question of whether sequences of selfish moves that are essentially chosen at random, do in fact also converge quickly to Nash equilibrium. We show in Section 1.4 that randomly-selected move sequences can be found in a distributed fashion, without any central control over agents. Let $w_{\max}$ denote the ratio of largest to smallest task weights. Our convergence rate bounds (Section 2) are the following:

1. (Section 2.1) The expected number of attempts made by RLS may be much lower than the maximal length of a sequence of self-improving moves, even allowing for unsuccessful attempts.

   For the class of problem instances that we use, RLS is seen to be very economical with basic operations of the form "evaluate the cost for task j of using resource ℓ". Essentially, we verify that useful progress is being made by randomly-chosen (typically suboptimal) moves.

2. (Section 2.2) We give an upper bound on the expected number of attempts for convergence by RLS that is polynomial in n, m and $w_{\max}$.

   This result shows that RLS does not get forced to make long sequences of moves of very small "value"; although such moves are occasionally necessary, we show that there exist enough opportunities to make moves that are (in a sense defined below) useful.

3. (Section 2.3) For integer task weights in the range $\{1, \ldots, w_{\max}\}$, for $w_{\max} \leq n$, we give a convergence rate of $O(m^2 n w_{\max})$.

The linearity in n is significant since the second part of Theorem 1.1 then says that in the distributed setting, the convergence rate would be independent of n; the $\Theta(n)$ speedup being a consequence of having n agents all working independently, in that model.

4. (Section 2.4) We give a $\Omega(n^2)$ lower bound on the expected number of attempts when task weights are unrestricted.

   This lower bound (as a function of n) is in contrast to the upper bound above; the construction uses a $w_{\max}$ that is exponential in n.

Note that the motivation for these results is that RLS may take place without central control over the agents; it is already known how to find Nash equilibria via a fairly simple algorithm. In particular Lemma 1 of Feldmann et al. [8] shows that given any initial assignment A, we may convert A to a Nash assignment using the following sequence of selfish steps. Let w_j be the weight of agent j, and assume that $w_1 \geq w_2 \geq \ldots \geq w_n$. Transfer w_1 to its optimal resource, then w_2, and so on through w_n (i.e. in descending order of task weight). In particular, they show that for uniform resources, a selfish move by an agent that selects its best resource, cannot cause another agent with higher weight to become unsatisfied. Here of course we do not have the guarantee that a big task will not become unsatisfied; instead most of our proofs use a potential function that is a special case of the potential function introduced recently by Even-dar et al. [7].

1.3 Related Work

Randomized Local Search has been studied in its capacity as a generic optimization method [19, 14, 18, 11]. Given a class of solutions that can be described by binary strings of length n, together with a function $f : \{0,1\}^n \to \mathbf{R}$ to be optimized, the approach is to maintain an n-digit binary string, and repeatedly flip randomly-chosen bits, accepting the change whenever f is higher for the modified string. (In the context of minimum spanning tree [14] and maximum matching [11] a variant is used where 2 bits may be flipped simultaneously.) In our setting, an assignment is an element of $[m]^n$ and we proceed by repeatedly replacing a random entry of this vector by a random element of $[m]$, and accept the new assignment if its potential is lower. RLS is a variant of the more extensively studied (1+1) Evolutionary Algorithm of [6], and for f we are using the potential function from [7].

The resource-sharing problem studied here was introduced by Koutsoupias and Papadimitriou [12]. It is shown in Fotakis et al. [9] how to find pure Nash equilibria in polynomial time. The term "Nashification" was introduced in Feldmann et al. [8] to describe the problem of finding a sequence of moves from a given assignment to a Nash assignment. For resources that may have variable capacities, they show how to find in polynomial-time, a sequence of moves that leads to a Nash equilibrium, where no move increases the social cost but moves are not necessarily self-improving. For uniform resources, [8] also gives an exponential lower bound on the number of self-improving moves that can be taken if "bad" choices are made. Even-dar et al. [7] show (for uniform resources) that the number of moves made by an algorithm that always moves the lowest-weight task, may be exponential. Our upper bounds on convergence rate are saying that for polynomial $w_{\max}$, a randomly-selected move sequence should avoid this worst-case behaviour. (Technically, we have not ruled out the possibility that no long move sequences exist for polynomial $w_{\max}$, but we show that for a suitable polynomial number of moves, there must exist an opportunity for the potential to go down by an appreciable amount, in a single move.)

Gairing et al. [10] consider the problem of finding and approximating the best and worst Nash equilibria, and also the computational complexity of finding bounded-length sequences of moves that lead to a pure Nash equilibrium (and show that it is hard to find a minimum length path). Even-dar et al. [7] also study the problem of finding sequences of self-improving moves that lead to a Nash equilibrium. The algorithms they consider include "Random", in which at each step a random task is selected, and moves to the lowest-cost resource (the *best response* strategy). They obtain a bound of $O(n^2)$ on the time taken for it to converge, for uniform resources.

There is a relationship with load balancing, in that the tasks can be treated as balls and the resources as bins, and if tasks as well as resources are uniform, then the objective is to distribute them evenly. Various simple randomized algorithms have been studied in this context. Azar et al. [1] studies a process where each ball is selected in turn, along with a bin selected from a sample of d bins, and the ball is placed in the bin

that has lowest contents amongst the sample of d. Vöcking [17] studies a variant where the sample of d is not necessarily uniform (and selection of elements of that sample may depend on each other.) The process we study (in the special case of uniform tasks and resources) corresponds to repeatedly selecting a ball at random, selecting a bin at random (use $d = 1$), and moving the ball to the bin if the load is less. We also mention work-stealing [2] as a related topic, in that load is being transferred from resources to less highly loaded resources, although in a different manner from [2], where resources pass load between themselves, whereas here it is units of load that seek out less loaded resources.

1.4 A Distributed Version of Randomized Local Search

The model is as follows. Each agent (independently of the others) makes a sequence of attempts using its own copy of Algorithm 1. If we assume a continuous time model, the process generates a sequence of events in which a single agent updates its strategy (i.e. choice of resource), and in the process changes the costs that may be experienced by other agents. We assume that the costs are updated without any delay. We use a continuous time model for simplicity, with agents operating asynchronously.

Algorithm 1: used by agent j, for $1 \leq j \leq n$

1. From initial assignment A, let $\ell = M(j, A)$, the resource assigned to j in A.
2. Repeatedly do the following:
   (a) Generate a delay δ from an exponential distribution $P(\delta) = \exp(-\delta)$; wait δ time units
   (b) Select resource ℓ' uniformly at random
   (c) evaluate the cost of using ℓ' instead of ℓ; if it is lower, replace ℓ by ℓ'.

Figure 1: Local algorithm, used by each agent

Comments: Using Algorithm 1, all agents behave the same; the weight w_j of agent j does not affect its behaviour, and they only interact via the effects they have on the costs of resources. If agent j selected the best ℓ' rather than a random ℓ' (the standard "best response" move), that would be a $\Theta(m)$ operation, and we wish to avoid the issue of a resource's cost being changed (by a different agent) in between being tested and being selected.

Algorithm 2: Randomized Local Search

1. Let A be an initial assignment.
2. Repeatedly do the following:
   (a) Select a task j uniformly at random
   (b) Select a resource ℓ uniformly at random
   (c) Assign ℓ to j if the potential (from [7]) of the resulting assignment is lower as a result.

Figure 2: Randomized Local Search

We show that Algorithm 1 is a distributed variant of Algorithm 2, in a strong sense.

Theorem 1.1 *Given any initial assignment A, if Nash assignment A' is found with probability $p(A')$ by Algorithm 1 then $p(A')$ is also the probability that Algorithm 2 finds A'.*

Let τ_1 be the expected number of steps taken for Algorithm 2 to find a Nash equilibrium. The expected time taken by Algorithm 1 is τ_1/n.

Proof: The significance of the exponential distribution is that it is "forgetful" in the sense that if $P(\tau) = \exp(-\tau)$ then for any positive τ_0:

$$P(\tau \mid \tau \geq \tau_0) = \exp(\tau - \tau_0) \quad \text{for } \tau \geq \tau_0$$

Consequently, at any point in time for n agents all using Algorithm 1, the next agent to make an attempted move will be selected uniformly at random, regardless of previous events.

For assignments A_1, A_2, let $p_1(A_1, A_2)$ be the probability that using Algorithm 1 on assignment A_1, the next usage of step (2c) by an agent replaces A_1 by A_2. Let $p_2(A_1, A_2)$ be the probability that using Algorithm 2 on assignment A_1, the next usage of step (2c) replaces A_1 by A_2.

Given any assignment A_1, the first agent using Algorithm 1 is selected uniformly at random, and itself selects a new resource uniformly at random, so for any A_1, A_2,

$$p_1(A_1, A_2) = p_2(A_1, A_2)$$

For assignment A let $D_i(A)$ (resp. $D'_i(A)$) be the probability distribution over assignments after i attempted moves by Algorithm 1 (resp. i steps of Algorithm 2). We have $D_{i+1}(A) = \sum_{A'} D_i(A')p_1(A', A)$, and $D'_{i+1}(A) = \sum_{A'} D'_i(A')p_2(A', A)$. By induction, $D_i(A) = D'_i(A)$.

Regarding the second part of the statement of the theorem: the expected value of exponential random variable δ is 1. Hence, during a time interval of length τ, the expected number of attempted moves made by an agent using Algorithm 1 is τ, and the expected number of attempted moves made by n agents is $n\tau$. This gives the ratio of n between its expected time and the expected number of steps of Algorithm 2. □

2 Convergence Rate Bounds

In this section we prove the bounds from Section 1.2 for the convergence rate of RLS.

Definition 2.1 *[7] Let A be an assignment. Define the real-valued function P as:*

$$P(A) = \sum_{\ell \in [m]} \frac{1}{c_\ell} (L(\ell, A))^2 + \sum_{j \in [n]} \frac{w_j^2}{c_{M(j,A)}}$$

Lemma 1 of [7] shows that if a task with weight w moves from resource ℓ to ℓ', then, letting A and A' denote the old and new assignments, we have that the potential P decreases by $2w(C(\ell, A) - C(\ell', A'))$. This is of course a positive quantity for any self-improving move.

In the case of uniform resources, we may simplify this function by putting $c_\ell = 1$ and removing its right-hand term which is constant. It is also convenient for our analysis to subtract off the minimum value that may be taken by $P(A)$ for any assignment A.

Definition 2.2 *Let A be an assignment for a problem instance with uniform resources.*

$$P(A) = \sum_{\ell \in [m]} (L(\ell, A))^2 - P^*$$

where P^* is the smallest value (for any assignment) that can be taken by $\sum_{\ell \in [m]} (L(\ell, A))^2$.

P is thus adjusted so that it has a minimal possible value of zero, although for a Nash assignment P may still be positive.

2.1 A Class of Problem Instances where RLS Improves on the Worst-case Move Sequence

We define a class of problem instances $\{\mathcal{I}_m : m \in \mathbf{N}\}$ that have the property that

1. The longest sequence of self-improving moves before reaching Nash equilibrium is $\Theta(m^3)$
2. The expected number of attempts made by Randomized Local Search is $o(m^3)$

Note that the expected number of attempts includes unsuccessful attempts, i.e. attempts where the resource selected by the task has too high a load for the task to benefit from moving to the selected resource.

In the following definition and proof, we will assume that m is a power of 2.

Definition 2.3 *$\mathcal{I}_m$ consists of m resources each with unit capacity and m^2 tasks each of unit weight. Let $A(m)$ denote an assignment for $\mathcal{I}_m$ in which resources $1, \ldots, m/2$ are each assigned $2m$ tasks and the remaining resources are assigned no tasks.*

Observation 2.4 *There is a unique Nash assignment for $\mathcal{I}_m$, namely the assignment of m tasks to each of the m resources.*

Proposition 2.5 *There exists a sequence of $\Theta(m^3)$ self-improving moves from $A(m)$ to the Nash assignment.*

Proof: Note that in the special case of uniform tasks and resources, $L(\ell, A)$ is just the number of tasks on resource ℓ in A.

Begin with the following sequence of moves. For $\ell = 1, \ldots, \frac{m}{4}$, move $\frac{m}{2} + \ell$ tasks from resource $\frac{m}{4} + \ell$ to resource $\frac{3m}{4} + 1 - \ell$. The resulting assignment $A'(m)$ has loads:

$$\begin{array}{lcll} L(\ell, A'(m)) & = & 2m & \text{for resources } \ell = 1, \ldots, m/4. \\ L(\ell, A'(m)) & = & \frac{3}{2}m - \ell & \text{for resources } \ell = \frac{m}{4} + 1, \ldots, m/2. \\ L(\ell, A'(m)) & = & \frac{3}{2}m - \ell + 1 & \text{for resources } \ell = \frac{m}{2} + 1, \ldots, 3m/4. \\ L(\ell, A'(m)) & = & 0 & \text{for resources } \ell = \frac{3m}{4} + 1, \ldots, m. \end{array}$$

Observe that for each resource $\ell \in \{1, \ldots, \frac{m}{4}\}$, a total of $\frac{m}{4}$ tasks on ℓ may each make a sequence of $\frac{m}{2} - 1$ self-improving moves along resources $\frac{m}{4} + 1, \ldots, \frac{m}{2}, \frac{m}{2} + 2, \ldots, \frac{3m}{4}$, then a further move into resource $\frac{3m}{4} + \ell$.

We have not reached the Nash assignment yet, but note that the number of moves above is already greater than $(m/4)^2 \cdot (m/2) = \Omega(m^3)$. □

We use the potential function of Definition 2.2, $P(A) = \sum_{\ell=1}^{m} L(\ell, A)^2 - m^3$. P is zero at the Nash assignment and positive otherwise. Note that for the assignment $A(m)$ defined above, $P(A(m)) = m^3$.

Definition 2.6 *Given an assignment A for $\mathcal{I}_m$ and a resource ℓ, the* surplus *of ℓ, denoted $s(\ell, A)$, is the quantity* $\max\{0, L(\ell, A) - m\}$*, and the* deficit *of ℓ, denoted $d(\ell, A)$, is the quantity* $\max\{0, m - L(\ell, A)\}$.

We show that $P(A)$ is given by

$$P(A) = \sum_{\ell} s(\ell, A)^2 + \sum_{\ell} d(\ell, A)^2 \tag{1}$$

Equation (1) can be derived as follows.

$$\begin{aligned} P(A) &= \sum_{\ell : s(\ell,A) > 0} (m + s(\ell, A))^2 + \sum_{\ell : d(\ell,A) > 0} (m - d(\ell, A))^2 + \sum_{\ell : s(\ell,A) = d(\ell,A) = 0} m^2 - m^3 \\ &= \sum_{\ell : s(\ell,A) > 0} m^2 + \sum_{\ell : s(\ell,A) > 0} (s(\ell, A))^2 + \sum_{\ell : s(\ell,A) > 0} 2ms(\ell, A) + \sum_{\ell : d(\ell,A) > 0} m^2 \\ &\quad + \sum_{\ell : d(\ell,A) > 0} (d(\ell, A))^2 - \sum_{\ell : d(\ell,A) > 0} 2md(\ell, A) + \sum_{\ell : s(\ell,A) = d(\ell,A) = 0} m^2 - m^3 \end{aligned}$$

Since the sum of the surpluses equals the sum of the deficits, the 3rd and 6th terms cancel. Note that the 1st, 4th and 7th terms total $\sum_\ell m^2 = m^3$, which cancels with the last term. We are left with Equation (1).
Notation: Let $s_{\max}(A)$ denote $\max_{\ell=1,\ldots,m}(s(\ell, A))$, the maximum surplus of a resource in A. Similarly let $d_{\max}(A) = \max_{\ell=1,\ldots,m}(d(\ell, A))$, the largest deficit of any resource.

Observation 2.7 *Let $(A_1, A_2, A_3, \ldots)$ be a sequence of assignments where A_{i+1} is obtained from A_i by an attempt. Then $s_{\max}(A_i)$ is a non-increasing function of i.*

Theorem 2.8 shows that the expected number of attempts by RLS on assignment $A(m)$ as defined in Definition 2.3 is $o(m^3)$. Note that in $\mathcal{I}_m$ (having m^2 tasks and m resources) there are m^3 possible queries of the form "what is the cost for agent f of moving to resource ℓ?" Consequently, as m increases, a decreasing fraction of such queries are needed to find the Nash equilibrium.

Theorem 2.8 *Given any assignment A for $\mathcal{I}_m$ for which $P(A) \leq m^3$ and $s_{\max}(A) \leq m$, the expected number of attempts is $o(m^3)$.*

Proof: Let A_1 be an assignment for $\mathcal{I}_m$ where $P(A_1) \leq m^3$ and $s_{\max}(A_1) \leq m$.

Suppose that $\mathcal{S} = (A_1, A_2, A_3, \ldots, A_N)$ is a sequence of assignments generated by starting from A_1 and generating A_i from A_{i-1} by a random attempt, and N is chosen such that A_N is the first occurrence in $\mathcal{S}$ of the unique Nash assignment. We are interested in the quantity $E(N) = E(|\mathcal{S}|)$ under the condition that A_1 has potential $\leq m^3$ and $s_{\max}(A_1) \leq m$.

Since $P(A_i)$ is a non-increasing function of i, $\mathcal{S}$ may be decomposed into $\mathcal{S}_1\mathcal{S}_2\mathcal{S}_3$ where $\mathcal{S}_1$ contains assignments A_i for which $P(A_i) > m^{2.05}$, $\mathcal{S}_2$ contains assignments A_i where $m^{2.05} \geq P(A_i) > m$ and $\mathcal{S}_3$ contains assignments A_i where $P(A_i) \leq m$. We upper bound the expected lengths of $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$.

Bounding the expected length of $\mathcal{S}_1$

Suppose that $P(A_i) = \lambda m^2$, $m^{0.05} < \lambda \leq m$. We will derive the following lower bound on the expected loss of potential due to an attempt on assignment A_i:

$$E(P(A_i) - P(A_{i+1})) \geq \frac{\lambda^2}{2m} \tag{2}$$

We obtain a lower bound on $E(P(A_i) - P(A_{i+1}))$ as follows. Let x be the number of resources with positive surplus and let y be the number of resources with positive deficit. Note that the largest deficit $d_{\max}(A_i)$ satisfies $d_{\max}(A_i) \leq m$. By Observation 2.7 we also have $s_{\max}(A_i) \leq m$. Due to Equation (1) we deduce

$$x \geq \lambda \quad \text{and} \quad y \geq \lambda \tag{3}$$

since otherwise the potential would be less than λm^2.

Each resource with a positive surplus has probability $> \frac{1}{m}$ of holding the task selected by an attempt. There are y resources with positive deficit, and any task on a resource with positive surplus may move to any resource with positive deficit. The expected loss of potential due to an attempt is lower bounded by $\frac{xy}{m^2}$ times the average loss of potential due to a move from a resource with positive surplus to a resource with positive deficit. That in turn is lower bounded by

$$2x^{-1} \sum_{\ell : s(\ell, A_i) > 0} s(\ell, A_i) + 2y^{-1} \sum_{\ell : d(\ell, A) > 0} d(\ell, A_i) = 2x^{-1} \sum_\ell s(\ell, A_i) + 2y^{-1} \sum_\ell d(\ell, A_i) = 2(x^{-1} + y^{-1}) \sum_\ell s(\ell, A_i)$$

where the last equality follows from $\sum_\ell s(\ell, A_i) = \sum_\ell d(\ell, A_i)$. Overall we have

$$E(P(A_i) - P(A_{i+1})) \geq \frac{xy}{m^2}(x^{-1} + y^{-1}) \sum_\ell s(\ell, A_i) \geq \frac{\lambda}{m^2} \sum_\ell s(\ell, A_i) \tag{4}$$

where the last inequality follows from (3).

Since $s_{\max} \leq m$ (by Observation 2.7),

$$\sum_\ell s(\ell, A_i) \geq \frac{1}{m} \sum_\ell s(\ell, A_i)^2.$$

Since $d_{\max} \leq m$,

$$\sum_{\ell} s(\ell, A_i) = \sum_{\ell} d(\ell, A_i) \geq \frac{1}{m} \sum_{\ell} d(\ell, A_i)^2.$$

Hence, from the above and Equation (1):

$$\sum_{\ell} s(\ell, A_i) \geq \frac{1}{2m} P(A_i).$$

From the above and (4) we deduce

$$E(P(A_i) - P(A_{i+1})) \geq \frac{\lambda}{m^2} \frac{1}{2m} P(A_i) = \frac{\lambda}{m^2} \frac{1}{2m} \lambda m^2 = \frac{\lambda^2}{2m}$$

as desired for (2).

> By way of an additional comment, we have been using $s_{\max} \leq m$, but the result could be extended to apply for higher $s_{\max}$ by showing that $s_{\max}/m$ is a lower bound on the expected loss of potential. A convenient case analysis divides $s_{\max} \geq m \cdot \lambda^{1/2}$ from $s_{\max} \leq m \cdot \lambda^{1/2}$; and you get an overall expected loss of potential of $\lambda^{1/2}$, which is not as good as what we have here, but still good enough to get the subcubic bound.

λ goes down from m to $m^{2.05}$, equivalently, P goes down from m^3 to $m^{2.05}$. We divide this range of P into $\leq \log m$ ranges:

$$\{[2^i m^2, 2^{i+1} m^2] \ : \ i = \log_2(m^{0.05}), \ldots, \log_2 m - 1\}$$

Over range i (of length $2^i m^2$), the expected loss of potential per attempt is, by (2) at least $(2^i)^2/2m$, so the expected length of the sequence of attempts over the i-th range is $2^i m^2 / ((2^i)^2/2m) = 2m^3/2^i$. This is maximised when $i = \log_2(m^{0.05})$, when the expected length of the sequence of attempts is $2m^{2.95}$. Over the whole range the expected number of attempts is $O(m^{2.95} \log m) = o(m^3)$.

Bounding the expected length of $\mathcal{S}_2$

We want to obtain a bound $o(m^3)$ on the expected number of attempts for the potential to go down from $m^{2.05}$ to m. To do so we show that for any assignment A_i with $m < P(A_i) \leq m^{2.05}$ that

$$E(P(A_i) - P(A_{i+1})) > \Theta(m^{-0.95}) \tag{5}$$

Let x be the number of resources ℓ with $s(\ell, A_i) \geq s_{\max} - 1$. We have the upper bound on x:

$$x(s_{\max}(A_i) - 1)^2 \leq P(A_i)$$

Let R^- be the set of resources with load $< m + s_{\max}(A_i) - 1$.

We obtain a lower bound on $E(P(A_{i+1}) - P(A_i))$ that is useful for $s_{\max}(A_i) \geq m^{0.14}$. If $s_{\max}(A_i) \geq m^{0.14}$, consider two cases: $|R^-| \geq m/4$ and $|R^-| < m/4$. In the first case, the expected loss of potential is lower bounded by the expected loss of potential due to a task on a resource with surplus $s_{\max}(A_i)$ moving to a resource in R^-, which is the probability that a task on a resource with surplus $s_{\max}(A_i)$ is chosen, times the probability that a resource in R^- is chosen, times twice the average difference between $m + s_{\max}$ and the load of a resource in R^- (whose average load is $< m$), hence

$$\geq \frac{1}{m} \cdot \frac{1}{4} \cdot 2m^{0.14} = \Omega(m^{-0.86})$$

In the second case, (since total surplus equals total deficit) we have the maximum deficit $d_{\max}(A_i) \geq s_{\max}(A_i)$ (since $\frac{3}{4}$ of resources have a surplus of $s_{\max}(A_i)$ or $s_{\max}(A_i) - 1$, and $s_{\max}(A_i) \geq m^{0.14}$.) Then the expected loss of potential is lower bounded by the expected loss of potential due to a resource on a task with surplus $s_{\max}(A_i)$ or $s_{\max}(A_i) - 1$ moving to a resource with deficit $d_{\max} \geq m^{0.14}$, which is

$$\geq \frac{3}{4} \cdot \frac{1}{m} \cdot m^{0.14} = \Omega(m^{-0.86})$$

Using a similar argument we may obtain the same bound on the expected loss of potential provided that $d_{\max} \geq m^{0.14}$. Define R^+ to be the set of tasks that have load $> m + 1 - d_{\max}(A_i)$, so R^+ is the set of all resources from which tasks may transfer to a resource with deficit $d_{\max}(A_i)$. If $|R^+| \geq m/4$ we have $m/4$ resources which between them have at least $m^2/4$ tasks, any of which may move to a resource with deficit $d_{\max}$ with an average loss of potential $\geq 2d_{\max}$. The expected loss of potential is

$$\geq \frac{1}{4} \cdot \frac{1}{m} \cdot 2m^{0.14} = \Omega(m^{-0.86})$$

If $|R^+| < m/4$ then at least a fraction $3/4$ of resources have deficit $d_{\max}$ or $d_{\max} - 1$, so $s_{\max} \geq d_{\max} = m^{0.14}$, and the previous case applies.

Next we obtain a bound that is useful when $s_{\max}(A_i) < m^{0.14}$ and $d_{\max}(A_i) < m^{0.14}$.

$P(A_{i+1}) - P(A_i) \leq 2(s_{\max}(A_i) + d_{\max}(A_i))$. $P(A_i) \geq m$ so we need at least $\geq (m/2)(s_{\max}(A_i) + d_{\max}(A_i))^{-1}$ moves to reduce it to 0; each resource may acquire (respectively, lose) $\leq d_{\max}(A_i)$ (respectively, $s_{\max}(A_i)$) tasks, so there are at least $m/2(s_{\max}(A_i) + d_{\max}(A_i))s_{\max}(A_i)$ resources with a surplus and at least $m/2(s_{\max}(A_i) + d_{\max}(A_i))d_{\max}(A_i)$ resources with a deficit.

The probability that an attempt finds a move satisfies

$$Pr(P(A_{i+1}) < P(A_i)) \geq \frac{1}{4(s_{\max}(A_i) + d_{\max}(A_i))^2 s_{\max}(A_i) d_{\max}(A_i)} \geq \frac{1}{16 s_{\max}(A_i)^3 d_{\max}(A_i)^3}$$

and when $P(A_{i+1}) < P(A_i)$, then $P(A_{i+1}) \leq P(A_i) - 2$ so the expected loss of potential is

$$\geq 1/8 s_{\max}(A_i)^3 d_{\max}(A_i)^3 = \Omega(m^{6 \times 0.14}) = \Omega(m^{-0.84}).$$

In all cases the expected loss of potential is at least $\Omega(m^{-0.86})$ which is sufficient for (5).

Bounding the expected length of $\mathcal{S}_3$

In this phase the potential is $P(A_i) \leq m$. Each move reduces the potential by ≥ 2. We lower bound the probability that a random attempt will result in a move, as a function of the potential P.

Case 1: Some resource ℓ has load $> m + 1$. Assume that ℓ has the maximum load. The probability that a task on ℓ is chosen is $> \frac{1}{m}$. The probability that a resource with load $\leq m$ is chosen is $\geq \frac{1}{2}$, since otherwise the potential would be $> m$ (there would be $> m/2$ resources with load $> m$, hence $> m/2$ tasks would have to move to obtain Nash equilibrium, and each of those moves must reduce the potential by ≥ 2.)

Hence the expected number of attempts until a move is $\leq 2m$.

Case 2: Some resource ℓ has load $< m - 1$, and no resource has load $> m + 1$. The probability that a task on some resource with load $\geq m$ is chosen in an attempt is $\geq \frac{1}{2}$, since in this case no resource has load $> m + 1$.

The probability that ℓ is chosen in an attempt is $\frac{1}{m}$. Hence the expected number of attempts until the next move is $\leq 2m$.

Case 3: All loads are either $m + 1$, m or $m - 1$.

Observe that any move will reduce by 1 the number of resources with a load of $m + 1$, and similarly the number of resources with a load of $m - 1$. Let k be the number of resources with a load of $m + 1$. We will obtain a bound N_k on the expected number of attempts to reduce the number of resources with load $m + 1$ (and similarly $m - 1$) from k to $k - 1$. Then the expected number of attempts overall is upper bounded by $N_1 + N_2 + \ldots + N_m$.

There are $\geq k(m + 1)$ unsatisfied tasks, so probability $k(m + 1)/m > k/m$ that an attempt chooses an unsatisfied task. There is probability k/m that a resource with load $m - 1$ is chosen. Hence the attempt succeeds with probability $\geq k^2/m^2$. Hence $N_k \leq m^2/k^2$.

The expected number of attempts to reach the Nash assignment is

$$\sum_{k=2}^{m} \frac{m^2}{k^2} = m^2 \sum_{k=2}^{m} k^2 < m^2$$

If we are in Case 3, we have $O(m^2)$ bound on the expected number of attempts. If we are in Cases 1 or 2, then the expected number of attempts that occur before either the Nash assignment is reached, or else Case 3 is reached, is also $O(m^2)$. Overall we have the bound $O(m^2)$ on the expected number of attempts required to reduce the potential from m to 0.

□

2.2 An Upper bound for Convergence Time in Terms of Task Weight Range

We will say that an assignment A *admits* a move of value v if there exists some task that for assignment A can be moved so that the potential goes down by $\geq v$ (but we do not require that the next move actually reduces the potential by $\geq v$).

Proposition 2.9 *There is no sequence $(A_1, A_2, \ldots, A_{m^2n})$ of m^2n assignments having the following properties:*

1. *For $i = 1, \ldots, m^2n - 1$, A_{i+1} is obtained from A_i via a single move*
2. *For $i = 1, \ldots, m^2n$, A_i does not admit a move of value $\geq \frac{1}{4}$*
3. *$P(A_1) - P(A_{m^2n}) \leq \frac{1}{4}$*

Proof: Suppose that the above properties hold, and we prove that any sequence that satisfies them must have length $< m^2n$.

Define a set of labels $R_1, R_2, \ldots, R_m$; initally in assignment A_1, give resource ℓ the label R_ℓ, for $\ell = 1, \ldots, m$. Let $label(\ell, A)$ denote the label of resource ℓ in assignment A (so for $\ell = 1, \ldots, m$, $label(\ell, A_1) = R_\ell$). Now, suppose the i-th move (from A_i to A_{i+1}) consists of task t moving from resource α to resource β. When a move is made, we exchange the labels associated with α and β. Hence, $label(\beta, A_{i+1}) = label(\alpha, A_i)$ and $label(\alpha, A_{i+1}) = label(\beta, A_i)$. For each $\ell = 1, 2, \ldots, m$ and $i = 1, \ldots, m^2n$, let $L(R_\ell, A_i)$ denote the load of the resource having label R_ℓ in assignment A_i. Let $res(R_\ell, A_i)$ denote the resource having label R_ℓ in assignment A_i.

Suppose that the i-th move transfers task j from $res(R_s, A_i)$ to $res(R_t, A_i)$ (where s and t are resources). Before the move, we have

$$L(R_s, A_i) - L(R_t, A_i) > w_j \geq 1 \tag{6}$$

Note that because we exchange the labels of the resources that j moves between, the label of j's resource is R_s before and after the move. The loss of potential is given by

$$P(A_i) - P(A_{i+1}) = 2w_j \cdot (L(R_s, A_i) - L(R_s, A_{i+1})) \geq 2(L(R_s, A_i) - L(R_s, A_{i+1})) \tag{7}$$

Since resource capacities are uniform, $L(R_s, A_i) - L(R_s, A_{i+1}) = L(R_t, A_{i+1}) - L(R_t, A_i)$, hence similarly the loss of potential is lower bounded by

$$P(A_i) - P(A_{i+1}) = 2w_j \cdot (L(R_t, A_{i+1}) - L(R_t, A_i)) \geq 2(L(R_t, A_{i+1}) - L(R_t, A_i)) \tag{8}$$

By Equations 7 and 8, if $|L(R_\ell, A_{i+1}) - L(R_\ell, A_i)| = \epsilon$ for any label R_ℓ, then the loss of potential satisfies

$$P(A_i) - P(A_{i+1}) \geq 2\epsilon \tag{9}$$

From 9 and Property 3, we have

$$\text{for } \ell = 1 \ldots, m,\ 1 \leq i', i'' \leq m^2n,\ |L(R_\ell, A_{i'}) - L(R_\ell, A_{i''})| \leq \frac{1}{8} \tag{10}$$

since any label R_ℓ that violated the above would have been involved in a sequence of steps between i' and i'' that reduced the potential by $\geq \frac{1}{4}$.

By Equations 6 and 10,

$$\text{for all } i' \geq i,\ L(R_j, A_{i'}) - L(R_k, A_{i'}) \geq \frac{3}{4} \tag{11}$$

We claim that furthermore $L(R_s, A_{i'}) - L(R_t, A_{i'})$ is monotonically non-increasing for moves i' subsequent to move i, i.e.

$$\text{for all } i' > i,\ L(R_s, A_{i'}) - L(R_t, A_{i'}) \leq L(R_s, A_{i'-1}) - L(R_t, A_{i'-1}) \tag{12}$$

Suppose otherwise. This could only happen if in some $A_{i'}$ a task j' moves to $res(R_j, A_{i'})$ from some other resource $res(R', A_{i'})$, in which case $A_{i'}$ admits a move of value $> \frac{3}{4}$ consisting of j' moving from $res(R', A_{i'})$

to $res(R_k, A_{i'})$ contradicting Property 2, or alternatively if in some $A_{i'}$ a task j' transfers from $res(R_k, A_{i'})$ to resource $res(R', A_{i'})$, in which case,

$$L(R_k, A_{i'}) - L(R', A_{i'}) > w_{j'} \geq 1$$

and by the upper bound established on the cumulative fluctuation of the loads (Equation 10),

$$\text{for all } i'',\ L(R_k, A_{i''}) - L(R', A_{i''}) \geq \frac{3}{4}$$

So putting $i'' = i$, in A_i, the assignment where j transfers from $res(R_s, A_i)$ to $res(R_t, A_i)$, we see that A_i admits a move of value $> \frac{3}{4}$, namely the transfer of j from $res(R_s, A_i)$ to $res(R', A_i)$. This again contradicts Property 2, so we have established Equation 12.

Observe that for all assignments A in the sequence, a move of task j from $res(R_s, A)$ to $res(R_t, A)$ causes the statement *"*$L(R_s, A) - L(R_t, A) > w_j$*"* to change from true to false, and from Equation 12, the statement remains false. There are $< nm^2$ statements of the form *"*$L(R_s, A) - L(R_t, A) > w_j$*"*, and when they have been exhausted, there are no further moves possible that satisfy the conditions of Proposition 2.9. □

Observation 2.10 *It would be interesting to know whether Proposition 2.9 holds without Property 2.*

Theorem 2.11 *Suppose that there are m resources, each with capacity 1, and n tasks each with weight in the range $[1, w_{\max}]$. Then the expected number of attempts required for Nash convergence is $O(w_{\max}^2 m^4 n^5 \log(mn))$.*

Proof: Define a *move search* to be a sequence of attempts that has one of the following two properties:

1. The sequence has length $< 4mn\log(mn)$, and the last attempt is a move, and no previous attempts are moves
2. The sequence has length $4mn\log(mn)$ and no attempts are moves

We say that a move search is *successful* provided that one of the following properties hold:

1. The last element of the move search is a move
2. The assignment is unchanged throughout the sequence, but it is a Nash assignment

It can be verified that for any assignment A, a random move search that begins with A has probability $< 1/n^4m^4$ of not being successful. (If A is Nash, the move search must be successful, and if A is not Nash, the move search has probability $\geq \frac{1}{mn}$ of succeeding at each attempt. It is sufficient for the length s to satisfy $(1 - \frac{1}{mn})^s < 1/m^4n^4$, which is satisfied if $s \geq 4mn\log(mn)$.)

Define a *move sequence search* to be a sequence of $\leq n^2m^2$ move searches, where the $(i+1)$-st move search starts at the final assignment of the i-th move search, which has one the following two properties:

1. The sequence contains $< n^2m^2$ move searches, and ends at the first assignment that admits a move of value $\geq \frac{1}{4}$
2. The sequence contains n^2m^2 move searches.

We say that a move sequence search is successful provided that all move searches in the sequence are successful. It can be seen that for any starting assignment, the probability that a move sequence search is not successful is $< m^2n/m^4n^4 = 1/m^2n^3$. Clearly a move sequence search has length $\leq 4m^3n^2\log(mn)$.

For assignment A_1, we prove that with probability $> \frac{1}{nm}$ a sequence of $4m^3n^2\log(mn)$ attempts starting from A_1 has one of the following properties

1. The final assignment is a Nash equilibrium
2. the potential decreases by $\geq \frac{1}{4}$

Given an upper bound on the initial potential, this allows us to upper-bound the expected time taken for Nash convergence, given that the potential is always positive.

We proceed as follows. Conduct a move sequence search starting from A_1. With probability $\geq 1 - \frac{1}{m^2 n^3}$ the move search sequence is successful. Given that it is successful, we have reached an assignment that admits a move of value $\geq \frac{1}{4}$, or else, by Proposition 2.9, the only alternative is that a Nash equilibrium has been reached. The first case gives us a $\frac{1}{mn}$ probability of reducing the potential by $\frac{1}{4}$, on the next attempt. The expected loss of potential is $\geq \frac{1}{4mn}$.

Initially the potential is $\leq (nw_{\max})^2$ (the maximum possible value being attained when n tasks all of weight $w_{\max}$ share the same resource).

After $mn \cdot N$ move sequence searches, we have that with probability $\geq \frac{1}{2}$, either the potential has fallen by $\frac{1}{4}N$, or else a Nash equilibrium has been found. Putting $N = 4(nw_{\max})^2$, we get a bound of $mn \cdot 4(nw_{\max})^2 \cdot 4m^3n^2 \log(mn) = O(m^4n^5w_{\max}^2 \log(mn))$ on the expected number of attempts. □

2.3 An Upper Bound for Integer Weights

We give a better upper bound on the convergence rate, that applies to the special case where task weights belong to the set $\{1, 2, \ldots, w_{\max}\}$, where $w_{\max}$ is an integer. We use the potential function of Definition 2.2. When weights are integers, we have a convenient lower bound of 2 on the drop in potential resulting from any move, which is the main reason why we obtain a stronger result than the corresponding one from the previous section.

Observation 2.12 *Suppose that task weights are in the range $[1, w_{\max}]$. If a move transfers a task from a resource with load L to a resource with new load L', then the loss of potential is at least $L - L'$.*

Proof: The weight w of the task must be $< L - L'$, and the potential goes down by $2w(L - L' - w)$, which is $\geq L - L'$. □

Observation 2.13 *Given n agents with weights in the range $\{1, 2, \ldots, w_{\max}\}$ and m identical resources, if assignment A satisfies $P(A) \geq mw_{\max}$, then the probability that an attempt on A succeeds is $\geq (m^2 w_{\max})^{-1}$.*

Proof: There must exist resources ℓ, ℓ' with $L(\ell, A) - L(\ell', A) \geq w_{\max}$, since if all resources had loads within $w_{\max}$ of each other, this would imply $P(A) < mw_{\max}$.

Let ℓ be a resource with maximal load, ℓ' a resource with minimal load.

Since $L(\ell, A) - L(\ell', A) > w_{\max}$, all the tasks on ℓ are unsatisfied. Resource ℓ holds at least $L(\ell, A)/w_{\max}$ unsatisfied tasks; since $L(\ell, A) \geq n/m$, there are at least $n/mw_{\max}$ unsatisfied tasks on ℓ. We have probability $\geq 1/mw_{\max}$ that an attempt selects an unsatisfied resource on ℓ, and probability $\geq 1/m$ that the resource selected by the attempt will allow that task to move to it, so the success probability is $\geq (m^2 w_{\max})^{-1}$ □

A better lower bound can be obtained when the potential $P(A)$ is at least twice as high as in Observation 2.13. In particular we have the following

Lemma 2.14 *Given n agents with weights in the range $\{1, 2, \ldots, w_{\max}\}$ and m identical resources, if assignment A satisfies $P(A) \geq 2mw_{\max}$, then the probability that an attempt on A succeeds is $\geq (mw_{\max})^{-1}$.*

Proof: There must exist resources ℓ, ℓ' with $L(\ell, A) - L(\ell', A) \geq 2w_{\max}$, since if all resources had loads within $w_{\max}$ of each other, this would imply $P(A) < 2mw_{\max}$.

Let ℓ be a resource with maximal load, ℓ' a resource with minimal load.

$L(\ell, A) - L(\ell', A) > 2w_{\max}$ and all the tasks on ℓ are unsatisfied. Consider two cases. Case (1): $\geq m/2$ resources have load $> L(\ell, A) - w_{\max}$ and Case (2): $< m/2$ resources have load $> L(\ell, A) - w_{\max}$.

In Case (1), there must be $\geq m/2$ resources all of whose tasks are unsatisfied; at least $1/2$ the total task load is on those resources (if we pick the $m/2$ most overloaded resources), hence at least a fraction $1/2w_{\max}$ of the n tasks are unsatisfied, i.e. $\geq n/2w_{\max}$ tasks. Hence an attempt has probability $\geq 1/2w_{\max}$ of selecting an unsatisfied task, and probability $\geq 1/m$ of selecting a resource to which that task may move, hence probability $\geq 1/2mw_{\max}$ of success.

In Case (2), there are $\geq n/mw_{\max}$ unsatisfied tasks (as in the proof of Observation 2.13) on resource ℓ. An attempt has probability $\geq 1/mw_{\max}$ of selecting one of these tasks, and probability $\geq 1/2$ of selecting a resource to which they may move, hence probability $\geq 1/2mw_{\max}$ of success. □

The statement of the theorem limits the maximum weight to be at most n, the number of tasks. A contrasting lower bound is given in the next section for the case when the maximum weight is unrestricted.

Theorem 2.15 *Let A be an assignment for n tasks whose weights lie in the range $\{1, 2, \ldots w_{\max}\}$, where $w_{\max} \leq n$, and m resources all with capacity 1. Then the expected number of attempts for Nash convergence, starting from A, is $O(m^2 n w_{\max})$.*

Proof: Suppose that $\mathcal{S} = (A_1, A_2, A_3, \ldots, A_N)$ is a sequence of assignments where $A_1 = A$ and A_i is generated from A_{i-1} by a random attempt, and N is chosen such that A_N is the first occurrence in $\mathcal{S}$ of a Nash assignment. We are interested in the quantity $E(N) = E(|\mathcal{S}|)$. Similarly to the proof of Theorem 2.8, we break $\mathcal{S}$ down into the concatenation of three subsequences, $\mathcal{S} = \mathcal{S}_1\mathcal{S}_2\mathcal{S}_3$. $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$ are defined as follows. $\mathcal{S}_1$ is assignments A_i such that $2nm < P(A_i)$, $\mathcal{S}_2$ is assignments A_i satisfying $2mw_{\max} < P(A_i) \leq 2nm$, and $\mathcal{S}_3$ is assignments A_i satisfying $P(A_i) \leq 2mw_{\max}$. We obtain bounds on the expected lengths of these subsequences that add up to a bound on the expected length of $\mathcal{S}$.

Bounding the expected length of $\mathcal{S}_1$

Observe that for all A_i, $P(A_i) \leq w_{\max} n^2$.

Since $P(A_i) > 2nm$, for some resources ℓ, ℓ', $L(\ell, A_i) - L(\ell', A_i) > 2n$.

Since we assume that $w_{\max} \leq n$, then when any task of weight w transfers from ℓ to ℓ', the loss of potential is $\geq wn$. The total task load on ℓ is $\geq n/m$. Letting n_ℓ be the number of tasks on ℓ, the average weight of tasks on ℓ is $\geq n/mn_\ell$. Hence the expected loss of potential due to a random task on ℓ moving to ℓ' is $\geq n^2/mn_\ell$. To lower bound the expected loss of potential, consider moves of tasks from ℓ to ℓ'. The expected loss of potential in an attempt is

$$E(P(A_i) - P(A_{i+1})) \geq \left(\frac{n_\ell}{n}\right)\left(\frac{n^2}{mn_\ell}\right) \cdot \frac{1}{m} = \frac{n}{m^2}$$

The expected number of attempts to reduce the potential to $2nm$ from its initial value of $\leq w_{\max} n^2$ is $\leq w_{\max} n^2/(n/m^2) = w_{\max} nm^2$.

Bounding the expected length of $\mathcal{S}_2$

We have $2mw_{\max} < P(A_i) \leq 2nm$.

By Lemma 2.14, the probability that an attempt succeeds is $\geq 1/mw_{\max}$. Any move (successful attempt) lowers the potential by ≥ 2, so the expected loss of potential in an attempts is $\geq 2/mw_{\max}$. Hence the expected number of attempts during $\mathcal{S}_2$ is $\leq 2nm/(2/mw_{\max}) = nm^2 w_{\max}$.

Bounding the expected length of $\mathcal{S}_3$

We have $0 < P(A_i) \leq 2mw_{\max}$.

The probability that an attempt succeeds is $\geq 1/nm$ for any non-Nash assignment. Any move lowers the potential by ≥ 2, so the expected loss of potential due to an attempt is $\geq 2/nm$. Hence the expected number of attempts during $\mathcal{S}_3$ before a Nash assignment is reached is $\leq 2mw_{\max}/(2/nm) = nm^2 w_{\max}$.

We have obtained bounds of $O(nm^2 w_{\max})$ on the expected lengths of each of $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$, hence an upper bound of $O(nm^2 w_{\max})$ on the length of $\mathcal{S}$. □

2.4 Lower Bound for Unrestricted Task Weights

We can show that without any restriction on the ratio $w_{\max}/w_{\min}$, the time taken for Nash convergence may be superlinear in n, hence dependent on n in the distributed model. (The following construction uses the same problem instance as Theorem 10 of [7], in the context of a different algorithm.)

Theorem 2.16 *There exists an instance with two identical resources and n tasks for which the expected number of attempts for Nash convergence is $\Theta(n^2)$.*

Proof: Let R_1 and R_2 be two resources each with capacity 1.

Assume n is an even number (it will be seen that if n is odd the construction can be easily modified). Let the task set $\{T_1, T_1', T_2, T_2', \ldots, T_{n/2}, T_{n/2}'\}$ be defined as follows. T_i and T_i' both have weight $3^{n/2-i}$.

For this problem instance, we claim that any Nash equilibrium must have T_i and T_i' on different resources, for $i = 1, \ldots, n/2$. If T_1 and T_1' are on the same resource, say R_1, then either of them would want to move to R_2, since the total weight of all other tasks is less than the weights of T_1 and T_1'. When T_1 and T_1' are placed on distinct resources, a similar argument applies to T_2 and T_2', and so on.

Let A be the assignment that places all tasks on R_1. We show that the expected number of moves starting from A is $\Theta(n^2)$.

Let $\mathcal{S}$ denote a sequence of assignments generated by a sequence of attempts starting at A. We may write $\mathcal{S}$ as the concatenation of substrings $\mathcal{S} = \mathcal{S}_1\mathcal{S}_2 \ldots \mathcal{S}_{n/2}$ defined as follows.

Let $\mathcal{S}_1$ be the prefix of $\mathcal{S}$ that contains all assignments up to and including the first one where T_1 and T_1' are assigned distinct resources. For $i = 2, \ldots, \frac{n}{2}$, let $\mathcal{S}_i$ be the substring of $\mathcal{S}$ (possibly empty) that contains all assignments subsequent to $\mathcal{S}_1 \ldots \mathcal{S}_{i-1}$ up to and including the first one where T_i and T_i' are assigned different resources. Noting that

$$E(|\mathcal{S}|) = \sum_{i=1}^{n/2} E(|\mathcal{S}_i|) \tag{13}$$

we obtain upper and lower bounds on the values $E(|\mathcal{S}_i|)$.

An attempt has a probability $2/n$ of selecting T_1 or T_1', and a probability $\frac{1}{2}$ of attempting to move that task to R_2, which would cause the resulting assignment to be the final one in $\mathcal{S}_1$. Hence

$$E(|\mathcal{S}_1|) = n.$$

Next we lower bound $E(|\mathcal{S}_i|)$ for $i \geq 2$. $\mathcal{S}_i$ is empty if T_i and T_i' are already assigned distinct resources at the end of $\mathcal{S}_1 \ldots \mathcal{S}_{i-1}$. To obtain a linear (in n) lower bound on $E(|\mathcal{S}_i|)$, we first lower bound the probability that at the end of $\mathcal{S}_{i-1}$, tasks T_i and T_i' are assigned to the same resource. Consider 2 cases:

1. $|\mathcal{S}_1 \ldots \mathcal{S}_{i-1}| < n/2$. In this case there is a probability $> \frac{1}{8}$ that no attempts have involved T_i or T_i', hence they are still assigned to R_1.

2. $|\mathcal{S}_1 \ldots \mathcal{S}_{i-1}| \geq n/2$. In this case there is a probability $> \frac{1}{4}$ that at least one attempt has involved T_i and T_i'. In that case, there is a probability $\geq \frac{1}{4}$ that the final attempt involving T_i and T_i' led to them occupying the same resource. Consider that final attempt: at the point it is made there are 3 possibilities:

   (a) T_i and T_i' occupy the less loaded resource. In that case the attempt will leave them together.

   (b) T_i and T_i' occupy the more loaded resource. In that case, with probability $\frac{1}{2}$ the task that made the attempt stays on that resource.

   (c) T_i and T_i' occupy distinct resources. In that case there is a probability $\frac{1}{4}$ that the task on the more loaded resource is chosen, and it attempts to move to the other resource.

   We have a lower bound of $\frac{1}{16}$ on the probability that T_i and T_i' are on the same resource.

Subject to the condition that $\mathcal{S}_i$ is non-empty, its expected length is n (similarly to $\mathcal{S}_1$ we are waiting for an attempt that involves T_i or T_i', which must then select the other resource). Since $\mathcal{S}_i$ is non-empty with probability $\geq n/16$,

$$\text{for } i = 2, \ldots, n/2,\ E(|\mathcal{S}_i|) \geq \frac{n}{16}. \tag{14}$$

By Equations 13 and 14

$$E(|\mathcal{S}|) \geq \sum_{j=1}^{n/2} E(|\mathcal{S}_j|) = \Omega(n^2).$$

For the upper bound, note that $E(|\mathcal{S}_i|) \leq n$, hence $E(|\mathcal{S}|) \geq n^2$. □

A variant of the above construction with three resources each of unit capacity, has the feature that an adversary could find an exponentially long sequence of moves. Under randomized local search the expected number of attempts is still $O(n^2)$.

3 Conclusions

Our interest in Randomized Local Search is due to it being able to be realised by distributed agents without central control, as described in Section 1.4. Our upper bounds on runtime for RLS are generally higher than for existing algorithms. Despite that, RLS may in fact be quite competitive, which is suggested by Theorem 2.8, in which we have shown a class of problem instances (that can probably be generalized) in which the convergence rate is less than the number of tasks multiplied by the number of resources. In contrast, any algorithm that picks tasks in some order and makes "best response" moves, will move $\Theta(n)$ tasks, each of them using $\Theta(m)$ time to find the best move. We have shown that under some circumstances it is faster to pick a random resource and move to it if the cost is lower, than to search for the best resource.

References

[1] Y. Azar, A. Broder, A. Karlin and E. Upfal. Balanced Allocations. *Procs. of the 26th ACM STOC, pp. 593-602, (1994)*

[2] P. Berenbrink, T. Friedetzsky and L.A. Goldberg. The Natural Work-Stealing Algorithm is Stable. *to appear in SICOMP, copyright SIAM.* Preliminary version in FOCS'01.

[3] P. Berenbrink, L.A. Goldberg, P.W. Goldberg and R. Martin. Utilitarian Resource Assignment. *manuscript.*

[4] A. Czumaj and B. Vöcking. Tight Bounds for Worst-Case Equilibria. *Procs. 13th Annual Symposium on Discrete Algorithms*, SIAM, Philadelphia, PA (2002), pp. 413–420.

[5] A. Czumaj, P. Krysta, and B. Vöcking. Selfish Traffic Allocation for Server Farms. *Procs. 34th Annual Symposium on Theory of Computing (STOC)*, Montreal, Canada (2002), pp. 287–296.

[6] S. Droste, T. Jansen and I. Wegener. On the analysis of the (1+1) Evolutionary Algorithm. *Theoretical Computer Science 276, pp. 51-81, 2002.*

[7] E. Even-Dar, A. Kesselman and Y. Mansour. Convergence Time to Nash Equilibria. *Procs. of 30th International Colloquium on Automata, Languages, and Programming (ICALP), 2003.*

[8] R. Feldmann, M. Gairing, T. Lücking, B. Monien and M. Rode. Nashification and the Coordination Ratio for a Selfish Routing Game. *Procs. of 30th International Colloquium on Automata, Languages, and Programming (ICALP), 2003.*

[9] D. Fotakis, S. Kontogiannis, E. Koutsoupias, M. Mavronicolas and P. Spirakis. The Structure and Complexity of Nash Equilibria for a Selfish Routing Game. *Procs. of 29th International Colloquium on Automata, Languages, and Programming (ICALP)*, Malaga, Spain (2002), pp. 123–134.

[10] M. Gairing, T. Lücking, M. Mavronicolas, B. Monien and P. Spirakis. The Structure and Complexity of Extreme Nash Equilibria. *manuscript*

[11] Oliver Giel and Ingo Wegener. Evolutionary Algorithms and the Maximum Matching Problem. Procs. of the 20th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2003) pp. 415-426.

[12] E. Koutsoupias and C.H. Papadimitriou. Worst-Case Equilibria. *Procs. 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, Trier, Germany (1999), pp. 404–413.

[13] M. Mavronicolas, and P. Spirakis. The Price of Selfish Routing. *Proc. 33rd Annual Symposium on Theory of Computing (STOC)*, Crete, Greece (2001), pp. 510–519.

[14] F. Neumann and I. Wegener. Randomized Local Search, Evolutionary Algorithms, and the Minimum Spanning Tree Problem manuscript

[15] T. Roughgarden and É. Tardos. How Bad is Selfish Routing? *Proc. 41st Annual Symposium on Foundations of Computer Science (FOCS)*, Redondo Beach, CA (2000), pp. 93–102.

[16] T. Roughgarden. Many papers studying the cost of selfish routing in the flow-model are available at Tim Roughgarden's web page `http://www.cs.cornell.edu/timr/` (which also has information and summaries).

[17] B. Vöcking. How Asymmetry Helps Load Balancing. *Procs. 40th IEEE-FOCS* (New York, 1999) pp. 131-140.

[18] I. Wegener. Towards a Theory of Randomized Search Heuristics MFCS '2003, LNCS 2747, 125-141, 2003.

[19] I. Wegener and C. Witt. On the Optimization of Monotone Polynomials by Simple Randomized Search Heuristics. *Procs. of GECCO 2003*, LNCS 2723, pp. 622-633