Nonlinear Optimization of Parallel Cooperative Tasks

Marc Wagger[†]

Project advisor: Abba Gumel[‡]

Key words. Optimization, Cooperation, Parallelization

AMS subject classifications. 46N10, 65F50

Abstract. It is a near universal notion that there are tasks to be completed and actors who set out to complete them. Frequently, cooperation is an option as multiple actors can work on the same task concurrently. In this paper, we derive a process for calculating the optimal effective rate and optimal solution for a general case of a set of actors cooperating to complete a set of tasks, as well as explicit formulas for special cases of these problems. We further conjecture that these calculations yield the exact rate and solution in finite computing time, noting room for algorithmic improvement. The results of these calculations can be directly and practically used to allow actors in cooperative situations to work at their full potential.

1. Introduction.

Joe can paint a house in 3 hours, and Sam can paint the same house in 5 hours. How long does it take for them to do it together?

This exact problem featured in the movie *Little Big League* [2], and problems like this are often given to middle school students. The solution involves converting the figures of hours per house to rates of houses per hour, $\frac{1}{3}$ and $\frac{1}{5}$, adding them, and converting back into a time for a final answer of $\frac{15}{8}$ hours.

When there are m actors (e.g., painters) but only a single task (e.g., houses to be painted), the *optimal effective rate*, O_T - the rate associated with the fastest completion of the tasks by the actors - is the sum of the rates for each actor:

$$O_T = \sum_{i=1}^{m} r_i \tag{1.1}$$

where r_i is the rate at which actor i can complete the task.

When there is a single actor and n tasks, the optimal effective rate is the harmonic mean of the rates divided by n:

$$O_T = \frac{1}{\displaystyle\sum_{j=1}^{n} \frac{1}{r_j}} \tag{1.2}$$

where r_j is the rate at which the actor can complete task j.

These approaches do not translate to problems with any number of actors and any number of tasks. An illustrative example is as follows:

[†]Department of Mathematics, University of Maryland College Park (mwagger@terpmail.umd.edu).
[‡]Department of Applied Mathematics and Statistics, University of Maryland College Park (agumel@umd.edu).

A pair of painters is tasked with painting 2 houses. The first painter is able to paint the first house in 3 hours alone while the second can do so in 5 hours, and the first painter is able to paint the second house in 5 hours alone while the second can do so in 3 hours. What is the shortest possible time it can take them to paint both houses?

If we approach the problem having the painters paint each individual house in optimal time, and use formula (1.1) for each task, we get an answer of $\frac{15}{4}$ hours. If we have each painter individually paint half of both houses in optimal time, and use formula (1.2) for each actor, we get an answer of 4 hours. However, it can be clearly seen that if we have both painters each paint the house they paint more quickly, they will finish painting both in 3 hours, which is a faster solution than what the above formulae give.

I had been working on a project which involved finding O_T for a setup with $m = 6$ actors and $n = 6$ tasks, and I was unable to find a solution for $m, n \geq 2$ in any prior literature. This paper details my findings concerning the solution to these problems, the likes of which appear frequently in situations which require cooperation and have a clearly defined actor-task relationship. Notable applications include parallel computing, manufacturing, and project management. The results in this paper include a solution to the general case of this problem along with algorithms to compute its exact value and a corresponding *optimal strategy*, A_O for completing the tasks at that rate.

2. Setup of the General Problem.

A problem is defined by m actors, n tasks, as well as an $m \times n$ matrix R of strictly positive rates, defined so R_{ij} denotes the rate at which actor i can complete task j. The solution involves an $m \times n$ matrix A of non-negative resource allocations, defined so A_{ij} represents the proportion of their total resources (proportion of time, for the above examples) actor i allocates to task j.

The effective rate e_t for the completion of task t is defined as:

$$e_t = \sum_{i=1}^{m} R_{it} A_{it} \tag{2.1}$$

The entire set of the n tasks, denoted T, isn't considered completed until each individual task is, and since the tasks are worked on in parallel, the set of tasks is not considered completed until the task with the slowest effective rate is completed. As a result, the effective rate for the entire set of tasks is:

$$E_T = min_{t \in T} e_t \tag{2.2}$$

O_T, therefore, is defined as:

$$O_T = max_{A \geq 0} E_T \tag{2.3}$$

If A is such that $\exists t_1, t_2 \in T$ where $e_{t_1} \neq e_{t_2}$, we can define set $T_{min} \subset T$ as the set of all tasks t where $e_t = E_T$. We can then pick any actor k and for all tasks $p \notin T_{min}$ decrease A_{kp} by a value ε, then for all tasks $q \in T_{min}$ we increase A_{kq} by:

$$\frac{\varepsilon \left| T \setminus T_{min} \right|}{R_{kq} \sum_{t \in T_{min}} \frac{1}{R_{kt}}} \tag{2.4}$$

For some $\varepsilon > 0$, this operation yields a new A for which the old T_{min} is a strict subset of the new T_{min} while also increasing E_T from its previous value. From this we can see that the A for which $E_T = O_T$ has the property that $\forall t \in T, e_t = E_T$.

Using this property, we can set up a matrix equation $M\mathbf{x} = \mathbf{b}$ to describe the problem with matrix $M \in \mathbb{R}^{m+n \times mn}$, vector $\mathbf{x} \in \mathbb{R}^{mn \times 1}$ and vector $b \in \mathbb{R}^{m+n \times 1}$ as follows:

$$M = \begin{bmatrix} 1_{1\times n} & 0 & \cdots & 0 \\ 0 & 1_{1\times n} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1_{1\times n} \\ S_1 & S_2 & \cdots & S_m \end{bmatrix} \mathbf{x} = \begin{bmatrix} A_{11} \\ \vdots \\ A_{1n} \\ A_{21} \\ \vdots \\ A_{mn} \end{bmatrix} \mathbf{b} = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ E_T \\ \vdots \\ E_T \end{bmatrix} \tag{2.5}$$

$1_{1\times n}$ represents a matrix of ones, and $S_k \in \mathbb{R}^{n\times n}$ represents a diagonal matrix with $S_{k_{ii}} = R_{ki}$.

The first m rows state that the sum of each actor's resource allocations is 1, which must yield the optimal strategy as otherwise time is wasted, and the remaining n state that $\forall t \in T, e_t = E_T$. O_T is the maximum value of E_T for which $M\mathbf{x} = \mathbf{b}$ has a non-negative solution, and the optimal strategy is the $\mathbf{x}$ which solves the system when $E_T = O_T$.

2.1. Solution for a Special Case: 2×2. When $m, n = 2$ and R is non singular, the above setup yields an equation with $M \in \mathbb{R}^{4\times 4}$, and solving it gives the following equations for E_T.

When $|R| < 0$:

$$E_T = min\left(\frac{R_{21}(R_{12}+R_{22})}{R_{21}+R_{22}}, \frac{R_{12}(R_{11}+R_{21})}{R_{11}+R_{12}}\right) \tag{2.6}$$

When $|R| > 0$:

$$E_T = min\left(\frac{R_{11}(R_{12}+R_{22})}{R_{11}+R_{12}}, \frac{R_{22}(R_{11}+R_{21})}{R_{21}+R_{22}}\right) \tag{2.7}$$

These can be calculated to determine the maximum value for E_T, which then must be O_T. Further, the optimal strategy can be calculated by replacing E_T with O_T in (2.5) and solving the resulting system.

3. Calculating $\mathbf{O}_T fortheGeneralCase$. In the vast majority of cases, $M\mathbf{x} = \mathbf{b}$ represents an underdetermined system of equations. This makes directly calculating an optimal effective rate very difficult, especially for large m and n. However, since we want our solution $\mathbf{x}$ to be non-negative, we may apply the following lemma to make this process more direct.

Lemma 3.1 (Farkas [1]). *For matrix $M \in \mathbb{R}^{m+n\times mn}$ and vector $\mathbf{b} \in \mathbb{R}^{m+n\times 1}$ either $\exists\mathbf{x}$ non-negative satisfying $M\mathbf{x} = \mathbf{b}$, or $\exists\mathbf{y} \in \mathbb{R}^{m+n\times 1}$ satisfying both $M^T\mathbf{y}$ non-negative and $\mathbf{y}^T\mathbf{b} < 0$.*

The above lemma gives the following corollary which we may reframe the problem around:

Corollary 3.2. *Defining Y as the set of vectors $\mathbf{y} \in \mathbb{R}^{m+n}$ that satisfy $M^T\mathbf{y}$ non-negative, a non-negative solution to $M\mathbf{x} = \mathbf{b}$ exists if and only if $\mathbf{y}^T\mathbf{b} \geq 0$, $\forall\mathbf{y} \in Y$.*

Since $\mathbf{y}^T\mathbf{b}$ is linear for a fixed $\mathbf{y}$ with respect to E_T, there is a closed interval $J_\mathbf{y}$ on $\mathbb{R}$ representing the values of E_T which give $\mathbf{y}^T\mathbf{b} \geq 0$ for that $\mathbf{y}$. O_T is therefore the upper bound of $\bigcap_{\mathbf{y}\in Y} J_\mathbf{y}$, and equivalently the minimum of all upper bounds across all $J_\mathbf{y}$ where $\mathbf{y} \in Y$. The problem is set up in a way which necessarily guarantees a value for O_T, so the intersection of the $J_\mathbf{y}$ is non-empty.

Since each of the mn rows of M^T contains 2 positive components with the rest being 0, a distinct pair of one of the first m components and one of the last n components, it is clear that $\forall \mathbf{y} \in Y$, any negative values are contained either within the first m components or last n components of $\mathbf{y}$. If there were negative values among both sets of components, any row which has nonzero components in those places would yield a negative component in $M^T\mathbf{y}$.

We can break the remaining $\mathbf{y}$ into 3 cases:

Case 1: The sum of the last n components of $\mathbf{y}$ is non-negative.

Since each of the last n components of $\mathbf{b}$ are E_T, increasing E_T doesn't decrease $\mathbf{y}^T\mathbf{b}$. So, for $c \in J_\mathbf{y}$, it must hold that $c+1 \in J_\mathbf{y}$, meaning $J_\mathbf{y}$ has no upper bound.

Case 2: The last n components of $\mathbf{y}$ are non-positive and have a negative sum.

In this case, increasing E_T decreases $\mathbf{y}^T\mathbf{b}$, so the upper bound of $J_\mathbf{y}$ solves $\mathbf{y}^T\mathbf{b} = 0$. If we fix the last n components of $\mathbf{y}$, the E_T solving $\mathbf{y}^T\mathbf{b} = 0$ is minimized when the sum of the first m components is minimized while keeping $\mathbf{y}$ in Y, and this can be achieved by minimizing each component independently under this constraint. The minimum value of component $k \in [1, m]$ which results in components $n(k-1)+1$ to nk of $M^T\mathbf{y}$ being non-negative is $\max_{j\in[1,n]}(-R_{kj}\mathbf{y}_{m+j})$. The minimum of this value across all sets of last n components of $\mathbf{y}$ is the smallest upper bound of $J_\mathbf{y}$ across all $\mathbf{y}$ in this case.

Case 3: At least one of the last n components of $\mathbf{y}$ is positive, but their sum is negative.

Like Case 2, the upper bound of $J_\mathbf{y}$ solves $\mathbf{y}^T\mathbf{b} = 0$, and if we fix the last n components of $\mathbf{y}$, the first m components are minimized the same way, just with any negative values replaced with 0s. However, if we replace all of the last n components of a given $\mathbf{y}$ which are positive with 0, it retains the same minimizing first m components, but a smaller E_T now solves $\mathbf{y}^T\mathbf{b} = 0$. Such a new $\mathbf{y}$ can be found for all $\mathbf{y}$ in this case, and since the new $\mathbf{y}$s fall under Case 2, is clear that Case 2 gives a smaller smallest upper bound for a $J_\mathbf{y}$.

In accordance with Case 2, we can define $\mathbf{x} \in \{\mathbb{R}_*^+\}^n \setminus \{\mathbf{0}\}$ such that $\mathbf{x}_k = -\mathbf{y}_{m+k}$ giving:

$$O_T = \min_{\mathbf{x}\in\{\mathbb{R}_*^+\}^n\setminus\{\mathbf{0}\}} \left(\frac{\sum_{i=1}^{m} \max_{j\in[1,n]}(R_{ij}\mathbf{x}_j)}{\sum_{j=1}^{n} \mathbf{x}_j} \right) \tag{3.1}$$

4. Algorithm to Compute O_T.

O_T is the minimum over $\{\mathbb{R}_*^+\}^n \setminus \{\mathbf{0}\}$ of the following function:

$$f(\mathbf{x}) = \frac{\displaystyle\sum_{i=1}^{m} \max_{j\in[1,n]}(R_{ij}\mathbf{x}_j)}{\displaystyle\sum_{j=1}^{n}\mathbf{x}_j} \tag{4.1}$$

This function does not have continuous directional derivatives across all of S, so standard gradient descent is unable to be directly applied. Instead, properties of this function can be utilized to construct a descent algorithm to find the minimum.

If we consider a line in $\mathbb{R}^n$ which has some points with all non-negative components, we can define a real-valued parameter z for this line such that the components of points are linear with respect to z, and non negative if and only if $z \in [0,1]$ or $z \in [0,\infty)$. In either case, it can be observed that the function along this line, $f_z(z)$, is piecewise and continuous, and on each piece, $f_z(z) = \frac{az+b}{cz+d}$ with $\frac{d}{dz}f_z(z) = \frac{ad-bc}{(cz+d)^2}$. Since $cz+d$ is the sum of components of $\mathbf{x}$, it, and therefore$(cz+d)^2$, is positive on the relevant interval of z and both c and d remain constant throughout that interval. As z increases and a new linear function becomes maximal, the new a won't decrease and the new b won't increase, meaning $ad-bc$ will never decrease. All of this together means $\frac{d}{dz}f_z(z)$ will only go from negative to positive at most once (possibly being 0 on some interval) across the relevant interval of z.

This shows any local minimum of f has the same value as all other local minima of f and all these local minima together form a convex region in $\mathbb{R}^n$. Equivalently, if the parameterized line of z contains any points which minimize f, at least one of them is on a breakpoint of $f_z(z)$.

Further, no point with a component of 0 can be the minimum, and the minimum of f lies on the interior of its domain. None of the *max* functions can return 0 over $\{\mathbb{R}_*^+\}^n \setminus \{\mathbf{0}\}$ due to at least one component being positive, so setting a zeroed component to certain $\varepsilon > 0$ won't change the value of the numerator while increasing the denominator, decreasing f. These properties establish f as a *pseudoconvex* function of which we may find the global minimum by finding a local minimum on the interior.

f does not have continuous directional derivatives, and any discontinuities are where multiple arguments of a *max* function are equal. For each of the m *max* functions, a line l_i passing through the origin can be constructed such that all arguments in it are the same. The discontinuities of any directional derivative of f lie at all points which can be reached from the l_is with a negative linear combination of all but at least 2 of the standard basis vectors, as this construction generates all points where at least 2 arguments of one *max* function are equal.

A point can be shown to be the minimum of f if the directional derivative is non-negative in all directions. For an iterative descent process on this function, if a descent step starts at a point p which does not lie on a discontinuity in any directional derivative of f, only the derivatives in the standard basis directions need to be checked to determine if p is a minimum. If a directional derivative of f is not continuous at p, then any regions of discontinuity in the directional derivative - to be referred to as *breakregions* (see Figure 1) - on which p lies extend in linear combinations of some l_i and all but at least 2 standard basis vectors from p. In these cases, in addition to the standard basis directions, the derivative at p in all directions along any breakregions on which p lies must also be checked to determine if p is a minimum.

We can define a set D_p of directions consisting of the positive and negative standard basis directions, and, if p lies on the intersection of any breakregions, all positive and negative basis directions for these intersections. For any polytopic cone extending from p such that all points on any breakregion it contains lie on its boundary, D_p has been defined so any direction pointing through this cone from p can be expressed as a positive linear combination of directions in D_p which point through the cone from p. This property also holds for the "flat cone" (half of $\mathbb{R}^n$) and the "null cone" (all of $\mathbb{R}^n$). This behavior demonstrates that the directional derivative in any direction from p can be calculated by expressing that direction as a positive linear combination of directions in D_p and applying that same linear combination to the corresponding directional derivatives, and normalizing it. As a result, if all derivatives in the directions contained in D_p are non-negative at p, p minimizes f.

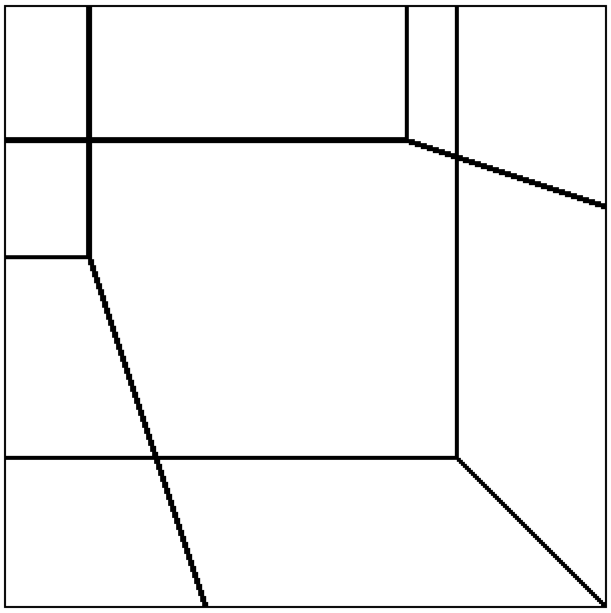

Figure 1. *Depiction of 2-D slice of $\mathbb{R}^3$ showing breakregions of f. Notice how the lines formed are either parallel to the axes or would pass through the origin (top left corner) if extended.*

4.1. Algorithm Outline. With these properties in mind, it can be seen that an iterative descent process to locate the minimum would only require checking the breakpoints of the $f_z(z)$s parameterized over the lines defined as containing the current point, c, and extending in each direction in D_c. The convex minimal area of f, along with the fact that the minimum of f lies on the interior of its domain give way to only needing to check the breakpoints of $f_z(z)$ instead of all critical points and endpoints to find the minimum. The breakpoint with the lowest value of f becomes the next point to descend from. If none of these breakpoints yield a lower value of f, O_T must be $f(c)$.

Algorithm 4.1 Descent to find O_T

```
Set $R$ = *rate matrix; elements present in $f$*
Set $p$ = *arbitrary point in* $\{\mathbb{R}_*^+\}^n$
while $p$ has been updated since the last iteration do
  for direction $d$ in $D_p$ do
    Consider parameterized function in that direction
    for j = 1, 2, $\cdots$ m do
      Identify breakpoint corresponding to *max* function $j$ of $f$
    end for
  end for
  Update $p$ to minimizing breakpoint across all directions in $D_p$
end while
$O_T = f(p)$
```

Conjecture (4.1): The above algorithm computes the exact value of O_T in finite time.

Because the involvement of directions which move along breakregions, and the strategy of jumping to breakpoints along these directions appears to eventually induce exact identification of the minimum. In practice, the above algorithm can encounter slow, oscillating descents which take a long, yet finite, time to resolve. It seems possible that the functions geometry could be further utilized to make the descent more efficient. It is possible, and perhaps likely, that an algorithm can be constructed which exactly computes O_T in low degree polynomial descent steps.

5. Computing $\mathbf{A}_O fortheGeneralCase$.

Considering system (2.5), the solution space moves linearly with respect to E_T, so this necessitates that when $E_T = O_T$, any non-negative solution to the system must not lie on the interior of the set of non-negative vectors. In other words, the smallest component of the vector solving the system corresponding to an optimal strategy must be 0. This can be achieved by descending over the space which is the solution to system (2.5) with $E_T = O_T$, with the aim of maximizing the smallest component of the vector, or the function:

$$f(\mathbf{x}) = min_{i<mn}\mathbf{x}_i \tag{5.1}$$

The dimensionality of the space to descend over is at least $mn - m - n$, but this can be reduced by considering properties of the optimal strategy. At least one of the components of all vectors corresponding to an optimal strategy must be 0, so we can add equations to the governing linear system of the form $x_a = 0$ with $a \leq mn$. To determine which equations preserve a nonnegative solution to the system when added, we can assess if the solveability criteria are still met once they are.

One way of doing this is to take the vector $\mathbf{y}$ which corresponds to a $\mathbf{p}$ minimizing the function f in (4.1) and consider matrix $B \in \mathbb{R}^{mn}$, defined with $B_{ij} = (M^T\mathbf{y})_{n(i-1)+j}$. Each row and column of B must contain a 0, as otherwise there is clearly room for components of $\mathbf{y}$ to change in a way which decreases the corresponding effective rate.

For some $\varepsilon > 0$, if we examine the result of an arbitrary perturbation of distance ε from $\mathbf{y}$ to $\mathbf{y}'$ (and define a corresponding B' as above), then if $\mathbf{y}'^T\mathbf{b}$ is negative with $E_T = O_T$, B' has negative components at indices where components of B are 0. This can be set up so that no perturbation of distance ε results in negative components of B' at indices where the components of B are nonzero. Adding rows to the system of equations which dictate that these components of a solution are 0 means only $\mathbf{y}'$ which satisfy $M^T\mathbf{y}' \geq 0$ without rows added to M can be the first $m+n$ components of any vector $\mathbf{y}'_*$ which satisfies $M^T\mathbf{y}'_* \geq 0$ with rows added to M.

This new system can be expressed as the matrix equation $M\mathbf{x} = \mathbf{b}$:

$$M = \begin{bmatrix} 1_{1\times n} & 0 & \cdots & 0 \\ 0 & 1_{1\times n} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1_{1\times n} \\ S_1 & S_2 & \cdots & S_m \\ K_{a_1} & & & \\ \vdots & & & \\ K_{a_b} & & & \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} A_{11} \\ \vdots \\ A_{1n} \\ A_{21} \\ \vdots \\ A_{mn} \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ O_T \\ \vdots \\ O_T \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{5.2}$$

defined as (2.5), with Ks being $1 \times mn$ matrices of all 0s and a single 1 at column a_b, where the a_bs are the indices of components in $M^T\mathbf{y}$ as defined above which are non-zero.

We can define a function g similar to f where $g(\mathbf{p})$ is the largest value of E_T for which a $\mathbf{y}_+$ with $(\mathbf{y}_*)_{m+k} = -\mathbf{p}_k$ satisfies Corollary 3.2 with M and $\mathbf{b}$ as defined in (5.2). The structure of g is such that its minimum represents the optimal effective rate under the new system, and, like f, it is pseudoconvex.

For a minimizing $\mathbf{p}$ of f, it can be seen that $g(\mathbf{p}) = f(\mathbf{p})$ as any perturbation to $\mathbf{y}_*$ of distance ε to $\mathbf{y}'_*$ which changes only the first m components of $\mathbf{y}_*$ is unable to result in both $M^T\mathbf{y}'_* \geq 0$, and $\mathbf{y}'^T_*\mathbf{b} = 0$ when $E_T < O_T$. This stems from the fact that these perturbations allow no $\mathbf{y}'_*$ which meet the second condition to also meet the first. This same argument demonstrates that $\forall \mathbf{y}'_*$ in the ε ball around $\mathbf{y}_*$, $g(\mathbf{p}') \geq f(\mathbf{p})$ for the corresponding $\mathbf{p}$ and $\mathbf{p}'$, and the pseudoconvexity of g means O_T is the absolute minimum value of both functions and the optimal effective rate in both scenarios.

In practical terms, this means components of the optimal solution A_O must be 0 when the corresponding component of B is not. Using the solution to system (5.2) as opposed to the solution to system (2.5) greatly reduces the dimensionality of the descent, resulting in quicker descent times. It can sometimes yield a single solution and remove the need for descent entirely.

If the maximizing $\mathbf{x}$ found through descent contains negative components or components greater than 1, these components will need to be manually adjusted to make the solution physically feasible. A_O can be approximated by setting $(A_O)_{ij} = \mathbf{x}_{n(i-1)+j}$.

6. Example Problem.

Letting

$$R = \begin{bmatrix} 3 & 1 & 4 \\ 1 & 5 & 9 \\ 2 & 6 & 5 \end{bmatrix} \tag{6.1}$$

the system of equations comes out to $M\mathbf{x} = \mathbf{b}$, where

$$M = \begin{bmatrix} 1 & 1 & 1 & & & & & & \\ & & & 1 & 1 & 1 & & & \\ & & & & & & 1 & 1 & 1 \\ 3 & & & 1 & & & 2 & & \\ & 1 & & & 5 & & & 6 & \\ & & 4 & & & 9 & & & 5 \end{bmatrix} \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ E_T \\ E_T \\ E_T \end{bmatrix} \tag{6.2}$$

and applying Algorithm 4.1 gives

$$O_T \approx 4.3902 \tag{6.3}$$

as well as

$$B \approx \begin{bmatrix} 0 & 15.14 & 12.83 \\ 3.79 & 0 & 0 \\ 0 & 0 & 6.1 \end{bmatrix} \tag{6.4}$$

which yields a system $M\mathbf{x} = \mathbf{b}$ for the optimal strategy of

$$M = \begin{bmatrix} 1 & 1 & 1 & & & & & & \\ & & & 1 & 1 & 1 & & & \\ & & & & & & 1 & 1 & 1 \\ 3 & & & 1 & & & 2 & & \\ & 1 & & & 5 & & & 6 & \\ & & 4 & & & 9 & & & 5 \\ & 1 & & & & & & & \\ & & 1 & & & & & & \\ & & & 1 & & & & & \\ & & & & & & & & 1 \end{bmatrix} \mathbf{b} \approx \begin{bmatrix} 1 \\ 1 \\ 1 \\ 4.3902 \\ 4.3902 \\ 4.3902 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{6.5}$$

the solution of which uniquely exists and corresponds to the allocation matrix of

$$A \approx \begin{bmatrix} 1 & 0 & 0 \\ 0 & .5122 & .4878 \\ .6951 & .3049 & 0 \end{bmatrix} \tag{6.6}$$

which describes the optimal strategy for this setup.

7. Application Scenarios. The following illustrate how to utilize the above techniques in scenarios which may not seem to fit with the problem setup:

In cases where the rates are variable, use of the expected value, or approximation thereof, of each variable rate may be used in the calculation of O_T. If a distribution of values for O_T is desired, running calculations on a sampling of possible rates may be checked to populate such a distribution.

In cases where the actors are only able to work on a task periodically, the rate for the actor and task should be multiplied by the proportion of time the actor is able to spend on the task.

In cases where actors have a chance of not being able to move a task towards completion, The rate for the actor and task may be multiplied by ($1-$that chance) for the calculation of the optimal effective rate. Alternatively, the probability of any task being unable to be completed may be calculated beforehand, and the value for O_T in cases where all tasks are able to be completed by all actors may be calculated.

In cases where an actor is entirely unable to move a task towards completion, the rate for the actor and that task may be set to a very small positive number. A rate of 0 may be used at one's own risk; the proofs involved in this paper bank on all rates being strictly positive, so having some be 0 runs the risk of the algorithm not converging to the correct value. From observation, the more sparse R is, the higher likelihood there is of this occurring.

If rates are being repeatedly approximated while the tasks are being completed, the new value for O_T may be calculated by taking the new approximated rates and multiplying them by the proportion of the corresponding task which is yet to be completed.

Figures for the rate at which O_T changes in response to one or multiple rates in R being changed can be derived experimentally by calculating the new value of O_T after the change.

8. Conclusion. By identifying solvability criteria for the problem, a formula for the optimal effective rate for the completion of a set of tasks by a set of actors was constructed. Descent algorithms to numerically calculate this rate as well as approximate the corresponding strategy for completing the tasks were also described. Establishing with certainty the speed and accuracy of these algorithms, or identifying algorithmic improvement are avenues for additional research. The version of the parallel cooperative task problem is idealized, so it may be of interest to investigate versions of this problem with penalties to go along with multiple actors working on a task at once or actors switching tasks. Regardless, the output generated by these algorithms are widely applicable in any situation where there is cooperation or parallelization, such as computing, manufacturing, and, of course, painting houses.

REFERENCES

[1] J. Farkas, *Theory of Simple Inequalities*, Journal of Pure and Applied Mathematics, 124 (1902), https://doi.org/https://doi.org/10.1515%2Fcrll.1902.124.1.

[2] A. Scheinman, *Little Big League*, 1994.