Runaway Task Assignment

F.M. Biermann[a,*], V. Naroditskiy[b], T.D. Nguyen[c], M. Polukarov[b], A. Rogers[b], N.R. Jennings[b]

[a] *International School of Economics at Tbilisi State University, Republic of Georgia*
[b] *School of Electronics and Computer Science, University of Southampton, UK*
[c] *Schools of Mathematics & Management, University of Southampton, UK*

Abstract

We analyse assignment problems in which not every agent is controlled by the central planner. The *free* agents search for vacant tasks autonomously, guided by their own preferences. The central planner, aiming to maximise the total value of the assignment, must take into account the behaviour of the uncontrolled agents. We model this situation as an $n+1$-player game played between n free agents and the central planner. We show that it is a weakly dominant strategy for the free agents to choose tasks according to their true preferences. Contrarily, the strategy of the central planner in the resulting Nash Equilibrium is highly complex – we prove that it corresponds to the solution of a mixed integer *bilevel* optimisation problem. Finally, we demonstrate how this program can be reduced to a computationally much more manageable disjoint bilinear program.

Keywords: assignment problem, stable matching, bilevel optimization

1. Introduction

Problems in economic theory are traditionally analysed in terms of stable outcomes (equilibria) or efficient solutions (optima). In the former case, the problem is considered in the context of the interaction of rational, self-interested, autonomous agents; in the latter, the agents are assumed to follow

*Corresponding author

Email addresses: `f.biermann@iset.ge` (F.M. Biermann), `vn@ecs.soton.ac.uk` (V. Naroditskiy), `T.D.Nguyen@soton.ac.uk` (T.D. Nguyen), `mp3@ecs.soton.ac.uk` (M. Polukarov), `acr@ecs.soton.ac.uk` (A. Rogers), `nrj@ecs.soton.ac.uk` (N.R. Jennings)

the instructions of the central planner who aims to optimise some objective. Of course, in realistic economic systems autonomous agents are often placed together with those controlled by the central planner, like public and private sectors jointly tackling social problems or locating economic activities. Typically, the autonomous agents will act to obtain their own individual goals, and the problem of the central planner is to coordinate the controlled agents so as to optimise the overall performance of the system, while taking into account the behaviour of self-motivated participants.

The present paper investigates a particular "semi-autonomous" scenario of this kind, namely *assignment problems* in which some of the players are autonomous and face private incentives to solve certain tasks. Instead of submitting to the planner's will, these agents strive to obtain the task that rates most highly according to their own preference rankings.

We call this situation *Runaway Task Assignment* (RTA) and model it as a game played between the central planner and the free agents. In RTA, the autonomous[1] agents, amended to the classical assignment problem, are assumed to have ordinal preferences over the available tasks. This arguably increases the robustness and applicability of our model. We neither require the central planner to form a belief about cardinal utility functions of the autonomous agents, nor do we assume the autonomous agents to be von Neumann-Morgenstern expected utility maximisers. In addition, adopting ordinal preferences allows us to directly utilise results from a branch of game theory, usually called *matching theory*, which originated with the seminal paper of Gale and Shapley (1962). From the start, matching theory evolved without drawing on the theory of expected utility.

The rest of the paper is organised as follows.

2. Runaway task assignment in the real world

Runaway task assignments arise naturally in the context of *location of economic activities.* In Koopmans and Beckmann (1957), for example, the authors discuss the assignment problem in the context of choosing locations for industrial plants under the standard assumption that the central planner is responsible for choosing the location for all of the plants. However, in

[1] We use the terms *free agents* and *autonomous agents* synonymously.

reality such tasks are typically divided between the public and the private sectors, where private businesses strive to maximise their own profits and the government is concerned with the overall welfare of the society. Note also that state institutions often have the priority over private entrepreneurs in making their choices, consistent with the way we define the RTA game.

As another example, consider *private-public partnerships* (PPP), where the public party, which usually supervises the complete project, intends to advance some public goal. In contrast, the participating private parties are primarily interested in those subprojects which have commercial potential. This poses an obstacle for assigning tasks in a globally optimal way. Companies will try to avoid those tasks which are unprofitable and difficult, trying instead to obtain subprojects promising high profits at low risk. A typical example is the provision of health care through hospitals and doctors, which is facilitated through private-public partnerships in many countries.[2] The payment agreements between the government and the private partners usually do not reimburse a hospital or doctor for *exactly* those costs associated with a specific patient. As a result, patients (= "tasks") yield different profit opportunities. Although hospitals/doctors (= "agents") participating in a PPP are not formally entitled to pick the profitable patients and reject the others, there may be informal ways to deter unprofitable patients.[3] The model presented in this paper could thereby prescribe an optimal policy for a public health system which both directly employs medical resources (doctors, hospitals etc.) and engages private contractors.

In the Internet economy, many *crowdsourcing systems* (see, e.g., Benkler (2006); Brabham (2008); Howe (2008)) can also be interpreted to be RTAs. In a crowdsourcing system, tasks which cannot satisfactorily be solved without human expertise are assigned to a group of more or less anonymous amateur problem solvers (the "crowd"). Yet companies making use of crowdsourcing do not have to totally rely on the crowd. For some of the tasks or even for all of them, they can engage professional problem solvers. These belong to their

[2]For an overview of private-public partnerships in the health sector, see Nikolic and Maikisch (2006).

[3]By entering "hospital turns away" or a similar phrase into an internet search engine, one gets plenty of media reports about exactly this issue. For example, UK dentists, working for the *National Health Service*, arguably behaved in such a way (Templeton (2007)). Reports about hospitals being reluctant to examine patients with X-ray or brain scans may straightforwardly be interpreted as avoidance of unprofitable tasks.

own personnel or a contractor's personnel who cannot reject tasks assigned to them. In contrast, crowd members can freely choose which tasks to work on, and they are probably not indifferent between all tasks. Hence, the firm has to find an optimal way of distributing its tasks between professional and amateur problem solvers.

Disaster response situations, providing prominent examples of crowdsourcing, also belong to this category. Consider a disaster relief situation where professional disaster responders coordinated by the government are assisted by local residents and disaster survivors. The government has neither the communication capabilities nor the authority to tell local participants what to do. However, local participants are very helpful and their efforts should not be ignored. Assuming the government can estimate the preferences of local participants (e.g., they prefer tasks closer to their current location to those further away), our work provides a way for the government to assign professional disaster responders optimally.

Situations resembling RTAs frequently occur in military campaigns. The 2011 war in Libya was fought by a coalition of NATO and loosely organised rebel troops who jointly tried to overthrow the regime of dictator Muammar Gaddafi. While the NATO forces were totally coordinated, it was arguably difficult to coordinate the actions of the rebels, who were untrained, unprofessional, and lacked command chains. Consequently, the NATO, as the central planner of the RTA, had to anticipate the prospective actions of the rebels when making its decisions on air strikes. Information about the rebels' next steps was provided by so called *liaison officers* (NATO representatives assigned to the rebel units).[4]

[4]Autonomous task choice can even be observed *within* military organisations, which are otherwise famous for their strict adherence to the principle of obeying orders. In military history it regularly occurred that ambitious commanders tried to gain fame by acting more bravely or by taking greater risks than desired by the central command. An outstanding example is the celebrated Danish naval officer Peter Jansen Wessel (1691-1720), called *Tordenskjold* (Danish for "thunder shield"). He constantly strived for the most prestigious tasks in the *Great Northern War* (1700-1721), thereby notoriously disobeying orders. His confrontation with the Swedish fleet in the *Battle of Dynekilen* (1716) in which his 7 ships captured 31 Swedish ships and destroyed another 13, was not backed by orders of the admiralty. Wessel's anarchistic conduct evoked considerable criticism in the Danish admiralty, eventually leading to a trial at a court-martial. Yet he was acquitted and even made an admiral later. His disobedience yielded huge personal prestige, as can be seen from the fact that Wessel is praised in the national anthems of both Denmark and Norway

3. Model

The *Runaway Task Assignment* game is a tuple

$$(F \cup \{cp\}, C, T, v, \succeq_{\boldsymbol{F}}, \boldsymbol{S}),$$

where $F \cup \{cp\}$ is the player set, consisting of n free agents in the set F and one central planner cp. C is the set of *coordinated agents* controlled by cp. We call $A := F \cup C$ the set of *agents*. T is the set of *tasks* of the assignment problem. An *outcome* of the game is a set $\mu \subseteq A \times T$ such that if $(a,t),(a',t') \in \mu$ it must hold $a \neq a'$ and $t \neq t'$. The function $v : A \times T \longrightarrow \mathbb{R}_+$ assigns to each agent-task pair a positive value, its *contribution*. The central planner prefers outcome μ for outcome μ' if

$$v(\mu) := \sum_{(a,t)\in\mu} v(a,t) > \sum_{(a,t)\in\mu'} v(a,t) := v(\mu').$$

In order to draw on matching theoretic results, we need to assume that whenever $\mu \neq \mu'$, then $v(\mu) \neq v(\mu')$. In this way, we ensure that two different outcomes are never equally desirable from the point of view of the central planner. $\succeq_{\boldsymbol{F}}$ is a *preference profile* which contains for each free agent $f \in F$ a linear[5] preference order $\succeq_f$ defined over a set $\mathcal{T}_f \subseteq T$.[6] The tasks in $\mathcal{T}_f$ are interpreted to be those which can in principle be accomplished by f. $\boldsymbol{S}$ is an *action set profile*, containing for each free agent f a set S_f of actions available to him and one action set S_{cp} of the central planner.

3.1. The action sets

The behaviour of the free players, i.e. the way in which they allocate themselves to tasks, is crucial for our model. We are going to specify their search process in this subsection. For simplicity, in what follows we will assume that there are no ties in the contributions of pairs and the values of matchings:

$$(a,t),(\hat{a},\hat{t}) \in A \times T: \quad (a,t) \neq (\hat{a},\hat{t}) \Rightarrow v((a,t)) \neq v((\hat{a},\hat{t})) \tag{3.1}$$

(the country he originated from). For an account of his deeds, see Chapter 1 ("A Knight Errant of the Seas") in Riis (2007).

[5] A linear ordering is total, transitive, and antisymmetric.

[6] From $\succeq_f$, the strict order $\succ_f$ and the indifference order $\sim_f$ are derived by the standard rules, i.e. $t \succ_f t' \Leftrightarrow t' \not\succeq_f t$ and $t \sim_f t' \Leftrightarrow t \succeq_f t' \wedge t' \succeq_f t$.

and

$$\mu, \mu' \in \boldsymbol{\mu} : \mu \neq \mu' \Rightarrow v(\mu) \neq v(\mu'), \tag{3.2}$$

with $v(\mu) := \sum_{(a,t)\in\mu} v((a,t))$.

We assume that the search process of the free players proceeds as follows. After the coordinated agents were assigned to tasks by the central planner, each free agent f approaches the task $t := \max_{\succeq_f} \mathcal{T}_f$. Note that due to antisymmetry of $\succeq_f$, this task is unique. If f finds t to be vacant, f takes over t. If f finds that a coordinated player already occupies t, f proceeds to the task which is second according to the preferences $\succeq_f$, namely $t' := \max_{\succeq_f} \mathcal{T}_f \setminus \{t\}$. Again, f checks the availability of t' and either takes it or continues with the subsequent item in its priority list. If there are no tasks left on f's priority list which were not yet approached, f stays idle. We summarize this as:

Behavioral Assumption 1. *Each free agent f approaches tasks according to a linear ordering $\succeq_f$ defined on a set $\mathcal{T}_f \subseteq T$.*

For two free players f' and f'' it may be the case that $\mathcal{T}_{f'} \cap \mathcal{T}_{f''} \neq \emptyset$. So what happens if f' and f'' approach the same task t? In this case, we assume that the agent better at performing the task, i.e.

$$\arg\max_{a\in\{f',f''\}} v((a,t))$$

keeps to t, while the other free agent continues the search process. This is a realistic assumption for scenarios in which free players, though being uncoordinated, have an interest in a high-valued solution of the problem (like in the disaster response application outlined in Section 2). We summarize this as:

Behavioral Assumption 2. *If a free agent f approaches a task t which is already occupied by a controlled agent, the search continues and the next site to be approached is*

$$\max_{\succeq_f}\{t' \in \mathcal{T}_f \mid t' \prec t\}$$

as long as there are sites in $\mathcal{T}_f$ which were not visited yet.

With these behavioral assumptions made for the free agents, the search process coincides with the Deferred Acceptance Algorithm of Gale and Shapley (1962) with men proposing, where:

- The free agents in F are the men and the tasksin T are the women.
- The men's preferences are given by $\succeq_{\boldsymbol{F}}$. All tasks which are not in the set $\mathcal{T}_f$ are considered unacceptable for f.
- The women's preferences are given by the valuation function v, i.e. for each $t \in T$ we have

$$f \succeq_t f' \Leftrightarrow v(f,t) \geq v(f',t). \tag{3.3}$$

  Together with (3.1), for each t this rule comprises a linear order $\succeq_t$ on F.
- Some tasks are blocked, namely those that are occupied by controlled agents.

We call this procedure the *Deferred Acceptance Algorithm with Blocked Tasks* (DAB).

3.2. Deferred Acceptance Algorithm with Blocked Tasks

The *Deferred Acceptance Algorithm* of Gale and Shapley (1962) constructs a stable matching in a *marriage market*. A marriage market is defined as a triple $(M, W, \succeq)$, where M is the set of "men" and W is the set of "women". A preference profile $\succeq$ maps each $m \in M$ into a linear preference order defined over $W \cup \{m\}$, and each $w \in W$ into a linear preference order defined over $M \cup \{w\}$ (the item x in x's preference order stands for the option of being single).[7] Naturally, from the fact that the deferred acceptance algorithm is finite and produces a unique output (Gale and Shapley (1962)), it follow that the DAB search process is finite and produces a unique output.

The order in which the free players propose to tasks, and the order in which they are rejected, does not influence the outcome assignment. This was shown by McVitie and Wilson (1971), who modified the original algorithm of Gale and Shapley (1962) so as to let men propose to women sequentially and in an arbitrary order (in Gale and Shapley (1962), the men propose simultaneously at each stage). They proved that the matching resulting from their algorithm is identical to the one generated by the standard deferred

[7]For a comprehensive discussion of marriage markets, see Roth and Sotomayor (1990), chapter 2.

acceptance algorithm. This finding of McVitie and Wilson (1971) implies that the outcome of the DAB algorithm is not affected by our assumption that the central planner assigns the coordinated agents first; in the DAB search process, the output matching would be the same even if the CP would assign the controlled agents when the free agents were already searching in the market. This is true as long as the coordinated agents could take away any task already occupied by a free agent, an assumption which is arguably reasonably for those applications we described in Section 2.

We define a *coordinated assignment* to be a matching $\mu_C \subseteq C \times T$ (no free player f is a member of any pair in μ_C). We denote by $(F, T, \succ_{\boldsymbol{F}})_{\mu_C}$ a marriage market formed by free agents and those tasks which are not matched under μ_C. Formally,

$$(F, T, \succeq)_{\mu_C} = (F, T \setminus \{t \mid (c, t) \in \mu_C\}, \succeq). \tag{3.4}$$

Here $\succeq$ is a preference profile which assigns to each $t \in T$ a linear order $\succeq_t$ according to (3.3) and to each free agent the order $\succeq_{\boldsymbol{f}}$. Given this, the set $\boldsymbol{\mu}^{\text{RTA}}$ consists of the following assignments:

Definition 1. *An assignment μ is* RTA-feasible *for a runaway task assignment $(C \cup F, T, v, \succ_{\boldsymbol{F}})$ if $\mu = \mu_F \cup \mu_C$ and the matching μ_F is the outcome of the DAB in the market $(F, T, \succ_{\boldsymbol{F}})_{\mu_C}$.*

4. Solution

In this section, we first transform the problem into a mathematical program. We then transform the program so that it becomes computationally manageable. Let binary variables x_{ij} indicate whether a controlled agent $i \in C$ is assigned to task $j \in T$, i.e. if $x_{ij} = 1$, then i is assigned to j, and if $x_{ij} = 0$, then this is not the case. Likewise, variables y_{ij} indicate whether a free agent i is matched to task j. We denote by $\boldsymbol{x}$ and by $\boldsymbol{y}$ matrices which have $|A|$ rows and $|T|$ columns and whose elements are either 0 or 1.

Theorem 1. *The solution to the Semi-Autonomuous Assignment problem*

$$(C \cup F, T, v, \succeq_{\boldsymbol{F}}),$$

coincides with the solution to the optimization problem

$$\max_{\boldsymbol{x}} \quad \sum_{(i,j)\in(C\times T)} v_{ij}x_{ij} + g(\boldsymbol{x}), \tag{4.1}$$

$$s.t. \quad \sum_{j\in T} x_{ij} \leq 1 \quad \forall i \in A \tag{4.2}$$

$$\sum_{i\in A} x_{ij} \leq 1 \quad \forall j \in T \tag{4.3}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in (A\times T) \tag{4.4}$$

where $g(\boldsymbol{x})$ is the value of the allocation of free agents to the tasks not assigned to controlled agents by $\boldsymbol{x}$, i.e.

$$g(\boldsymbol{x}) = \min_{\boldsymbol{y}} \quad \sum_{(i,j)\in(F\times T)} v_{ij}y_{ij} \tag{4.5}$$

$$s.t. \quad \sum_{j\in T} y_{ij} \leq 1 \quad \forall i \in F \tag{4.6}$$

$$\sum_{i\in F} y_{ij} \leq 1 - \sum_{i\in C} x_{ij} \quad \forall j \in T \tag{4.7}$$

$$y_{ij} + \sum_{k\succ_i j} y_{ik} + \sum_{l\succ_j i} y_{lj} \geq 1 \quad \forall (i,j) \in (F\times T) \tag{4.8}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in (F\times T) \tag{4.9}$$

Proof. First of all, we show that the solution of the mathematical program above corresponds to a *matching*. Constraint (4.2) requires each controlled agent to be assigned at most one task and constraint (4.3) requires each task to be allocated to at most one controlled agent. Likewise, constraint (4.6) requires each free agent to be assigned at most one task and constraint (4.7) requires each controlled task to be allocated to no free agent if it is already occupied by a controlled agent, or to at most one free agent otherwise. Constraints (4.4) and (4.9) ensure that all variables x_{ij} and y_{ij} are binary.
Next, we show that the matrix $\boldsymbol{y}$ determined in the solution of the program corresponds to the outcome of the DAB procedure when only those tasks are available which are not occupied by controlled agents. As mentioned, (4.7) ensures that no free agent is matched to a task assigned to a controlled agent. As we showed in Section 3, the DAB procedure converges to a stable matching in a marriage market $(F, T, \succ_{\boldsymbol{F}})_{\mu_C}$ (cf. (3.4) above). We now have

to show that the matrix $\boldsymbol{y}$ in a solution of the program above corresponds (1) to the stable matching in the market $(F, T, \succ_{\boldsymbol{F}})_{\mu_C}$, where μ_C is determined by the matrix $\boldsymbol{x}$, and (2) that this stable matching corresponds to *that* stable matching chosen by the free agents in the DAB search process.
For ensuring that $\boldsymbol{y}$ corresponds to a stable matching in the market $(F, T, \succ_{\boldsymbol{F}})_{\mu_C}$, we include the so called *blocking pair constraint* (4.8). This constraint is taken from Roth et al. (1993), who develop stable matching theory in a mathematical programming framework.
Finally, we have to show that the matching $\boldsymbol{y}$ derived from the solution of the above program is not just stable, but it is indeed the *same* stable matching as the one constructed through the DAB procedure. In DAB, free agents represent the proposing side, and the procedure converges to a stable matching that is *optimal* for the free agents: each agent prefers the optimal matching over any other stable matching (see Gale and Shapley (1962)). The optimal stable matching for the proposing side coincides with the *worst* stable matching of the responding side (Roth and Sotomayor (1990), Theorem 2.13 and Corollary 2.14, p. 33), which means that each task that is not occupied by a controlled agent prefers any other stable matching in the market $(F, T, \succ_{\boldsymbol{F}})_{\mu_C}$ over the matching selected through DAB. Let t be a task which is not occupied by a controlled agent and let (t, f^*) be the pair formed under the matching constructed through the DAB algorithm. Moreover, let $B(t) \subseteq T \times F$ be the set

$$B(t) := \{(t, f) \mid (t, f) \subseteq \mu, \mu \text{ is a stable matching in } (F, T, \succ_{\boldsymbol{F}})_{\mu_C}\}.$$

By definition of the "preferences" of the tasks (see (3.3) above) the fact that the constructed matching is the task-worst implies that

$$(t, f^*) = \text{argmin}_{(t,f)\in B(t)} v(t, f).$$

Verbally, if F_t denotes the set of free agents that perform task t in some stable matching in the market $(F, T, \succ_{\boldsymbol{F}})_{\mu_C}$, then, in the *free-agent-optimal* matching, task t is performed by the least-qualified among these agents—$\arg\min_{f \in F_t} v(t, f)$.
It follows that the objective function for the assignment of free agents, which corresponds to the matching chosen by the DAB algorithm in the market $(F, T, \succ_{\boldsymbol{F}})_{\mu_C}$, is given by (4.5). $\square$

The integrality constraint (4.9) can be relaxed as has been shown by Vande Vate (1989, Theorem 16), allowing to replace it with a nonnegativity constraint.

The SAAP can thus be fully specified as the following bilevel mixed integer linear program **SAAP(2LMILP)**:

$$
\begin{aligned}
\max_{\boldsymbol{x}} \quad & \sum_{(i,j)\in(C\times T)} v_{ij}x_{ij} + \sum_{(i,j)\in(F\times T)} v_{ij}y_{ij} \\
s.t. \quad & \sum_{j\in T} x_{ij} \le 1 \quad \forall i \in C \\
& \sum_{i\in C} x_{ij} \le 1 \quad \forall j \in T \\
& x_{ij} \in \{0,1\} \quad \forall (i,j) \in (C\times T) \\
& \boldsymbol{y} \text{ solves } \min \sum_{(i,j)\in(F\times T)} v_{ij}y_{ij} \\
& \qquad \sum_{j\in T} y_{ij} \le 1 \quad \forall i \in F \\
& \qquad \sum_{i\in F} y_{ij} \le 1 - \sum_{i\in C} x_{ij} \quad \forall j \in T \\
& \qquad y_{ij} + \sum_{k\succ_i j} y_{ik} + \sum_{l\succ_j i} y_{lj} \ge 1 \quad \forall (i,j) \in (F\times T) \\
& \qquad y_{ij} \ge 0 \quad \forall (i,j) \in (F\times T)
\end{aligned}
$$

Formally, we have a mixed integer *bilevel* optimisation problem—a hierarchical program in which the set of constraints contains a parametric optimisation problem. Solving bilevel programs is difficult in general, let alone solving one with binary variables, and applying known algorithms to the program at hand would yield solutions only for extremely small problem instances. The most popular method for solving bilevel programs is to replace the second level with a set of Karush-Kuhn-Tucker optimality conditions and then add these constraints to the first level to form a Mathematical Program with Equilibrium Constraints (MPEC) (Luo et al. (1996)). However, this introduces a set of complementary constraints that are difficult to deal with.[8] In fact, solving a linear bilevel program in which all functions are linear is

[8]The complementary constraints can then be transformed into a new set of constraints that involve integer variables using a Big-M method. Alternatively, nonlinear programming relaxation can be used to approximate these complementary constraints.

already strongly NP-hard (Marcotte and Savard (2005)).[9] In our case, the upper level contains binary variables and hence the problem is even more difficult.
For devising a way how to practically solve the SAAP problem, we will show that (**SAAP**(**2LMILP**)) is equivalent to a disjoint bilinear program, which is much more manageable computationally,[10] as stated in the following theorem:

Theorem 2. *The SAAP bilevel mixed integer linear programming model (***SAAP**(**2LMILP**)*) is equivalent to the following disjoint bilinear program* **SAAP**(**DBL**)*:*

$$
\begin{aligned}
\max_{\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\gamma}} \quad & \sum_{i\in C, j\in T} v_{ij}x_{ij} + \sum_{i\in F}\lambda_i + \sum_{j\in T}\beta_j(1-\sum_{i\in C}x_{ij}) + \sum_{(i,j)\in(F\times T)}\gamma_{ij} \\
s.t. \quad & \sum_{j\in T} x_{ij} \le 1 \quad \forall i\in C \\
& \sum_{i\in C} x_{ij} \le 1 \quad \forall j\in T \\
& x_{ij} \ge 0 \quad \forall (i,j)\in(C\times T) \\
& \lambda_i + \beta_j + \gamma_{ij} + \sum_{k\prec_i j}\gamma_{ik} + \sum_{l\prec_j i}\gamma_{lj} \le v_{ij} \quad \forall (i,j)\ \in (F\times T) \\
& \boldsymbol{\lambda}\le 0, \boldsymbol{\beta}\le 0, \boldsymbol{\gamma}\ge 0.
\end{aligned}
$$

At a high level, the transformation of (**SAAP**(**2LMILP**)) into a bilinear program involves three steps. First, we replace the linear program on the second level with its dual. Since the primal was a minimisation problem, the dual is a maximisation problem. Having maximisation in both first and second stages lets us combine the objectives and reduce the problem to a single-stage optimisation. The resulting problem belongs to the class of mixed integer non-convex quadratic programming problems and is still

[9]Even checking local optimality in linear bilevel programming is NP-hard, cf. Marcotte and Savard (2005).

[10]Although solving bilinear programs is still NP-hard (Audet et al. (1999)), the mathematical programming formulation is in a much nicer form, i.e. we only have to deal with a single minimisation problem instead of a minimax problem. Notice that not all linear bilevel programs can be transformed into a bilinear problem. However, a disjoint bilinear program can be transformed into a linear bilevel program.

quite difficult to solve. We then exploit the special structure of the problem to note that the integrality constraints on $\boldsymbol{x}$ can be dropped obtaining a bilinear program. The details follow in the proof.

Proof. Let λ_i, β_j and γ_{ij} be dual variables for constraints (4.6)-(4.8) for all $(i,j) \in (F \times T)$. The dual problem is formulated as:

$$
\begin{aligned}
\max_{\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\gamma}} \quad & \sum_{i\in F} \lambda_i + \sum_{j \in T}(1 - \sum_{i\in C} x_{ij})\beta_j + \sum_{(i,j)\in(F\times T)} \gamma_{ij} \\
s.t. \quad & \lambda_i + \beta_j + \gamma_{ij} + \sum_{k \prec_i j} \gamma_{ik} + \sum_{l \prec_j i} \gamma_{lj} \le v_{ij} \quad \forall (i,j) \ \in (F\times T) \\
& \boldsymbol{\lambda} \le 0, \boldsymbol{\beta} \le 0, \boldsymbol{\gamma} \ge 0.
\end{aligned}
$$

Plugging the dual into the original problem and combining two max operators, we obtain the following problem:

$$
\begin{aligned}
\max_{\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\beta},\boldsymbol{\gamma}} \quad & \sum_{i\in C, j\in T} v_{ij}x_{ij} + \sum_{i\in F} \lambda_i + \sum_{j\in T} \beta_j(1 - \sum_{i \in C} x_{ij}) + \sum_{(i,j)\in(F\times T)} \gamma_{ij} \\
s.t. \quad & \sum_{j\in T} x_{ij} \le 1 \quad \forall i \in C \\
& \sum_{i \in C} x_{ij} \le 1 \quad \forall j \in T \\
& x_{ij} \in \{0,1\} \quad \forall (i,j) \in (C\times T) \\
& \lambda_i + \beta_j + \gamma_{ij} + \sum_{k \succ_i j} \gamma_{ik} + \sum_{l \succ_j i} \gamma_{lj} \le v_{ij} \quad \forall (i,j) \ \in (F \times T) \\
& \boldsymbol{\lambda} \le 0, \boldsymbol{\beta} \le 0, \boldsymbol{\gamma} \ge 0.
\end{aligned}
$$

The objective function contains linear terms on $(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\gamma})$ and a bilinear term $-(\sum_{(i,j)\in(C\times T)} \beta_j x_{ij})$. Without this bilinear term, the problem will be equivalent to two separate optimisation problems: an assignment problem and a (dual of a) stable matching problem. Due to the presence of the bilinear terms together with the integrality constraint on x_{ij}, this problem belongs to the class of mixed integer non-convex quadratic programming problems and is quite difficult to solve. However, once we fix $(\boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\gamma})$, the objective function is linear on $\boldsymbol{x}$. The problem becomes:

$$\begin{aligned}
\max_{\boldsymbol{x}} \quad & \sum_{i \in C, j \in T} (v_{ij} - \beta_j) x_{ij} \\
s.t. \quad & \sum_j x_{ij} \leq 1 \quad \forall i \in C \\
& \sum_{i \in C} x_{ij} \leq 1 \quad \forall j \in T \\
& x_{ij} \in \{0, 1\}
\end{aligned}$$

This is an assignment problem,[11] and hence the integrality constraint can be relaxed (see for example Bertsimas and Tsitsiklis (1997, Corollary 7.2)). Thus, for *every* solution $\boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\gamma}$ (or equivalently, for every $\boldsymbol{y}$) to the agent-optimal stable matching problem, there is an integer solution $\boldsymbol{x}$ that is optimal. In other words, there is always an integer solution $\boldsymbol{x}$ that is optimal, and we can drop the integrality constraints in SAAP. This leads to the disjoint bilevel program **SAAP**(**DBL**). □

There is an extensive literature on bilinear programming. In particular, a simple greedy approach, such as 'hill climbing' (see, e.g., Russell and Norvig (2003)), may obtain high-quality solutions in little time. This is done by iteratively solving an LP for optimal $(\boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\gamma})$ for each fixed $\boldsymbol{x}$ and then solving another LP for an optimal $\boldsymbol{x}$ by fixing the newly found $(\boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\gamma})$. This process is repeated until the optimal value does not improve. At that point we obtain a locally optimal solution. It is noted also that the disjoint constraint sets in the SAAP problem are quite nice. The constraints on $\boldsymbol{x}$ define an assignment polyhedron with known extreme points and the constraints on $(\boldsymbol{\lambda}, \boldsymbol{\beta}, \boldsymbol{\gamma})$ correspond to a dual feasible space of the stable matching problem. This means the LP problems can be solved very efficiently and the algorithm converges very fast to a local optimal solution. White (1992) converts a bilinear program into a big LP whose constraints are generated sequentially through solving smaller LPs. This methods promises finite convergence and can be used to solve **SAAP**(**DBL**) as the assignment problem and the stable matching problem can be solved very efficiently. The bilinear program

[11] In the variant of the assignment problem stated here, the number of tasks may be different from the number of agents and tasks/agents may be left unassigned. One can convert this version to the standard assignment problem by adding dummy tasks/agents.

can also be reduced to a concave minimisation problem where an outter approximation algorithm can be applied (Thieu (1988)). More recent advanced methods for solving disjoint bilinear programming can be found in Alarie et al. (2001) who apply cutting plane methods to produce global optimal solutions. Alarie et al. (2001) show that cutting plane methods can be used to solve disjoint bilinear programming problems with up to 500 variables in each disjoint set and with 100 constraints.

5. Conclusions

Our work introduces assignment problems in which autonomous agents are placed together with those fully controlled by a central planner. The autonomous agents act to obtain their own individual goals. The central planner coordinates the controlled agents with the aim to optimise the overall performance of the system, while taking into account the behaviour of the self-motivated participants. This scenario resembles many economic situations, some of which were outlined in Section 2. SAAP belongs to models which combine rational and boundedly rational agents, which are rare in the game theoretic literature, though in reality many situations of this kind can be found.[12]

Clearly, the search process assumed for the free agents in SAAP is not the only reasonable model. Indeed, there are many other possibilities for how one could model the behaviour of the free agents. For example, many real-world scenarios could be better described with a stochastic search process. One might also consider search strategies taken from cognitive psychology, like the famous *satisficing* heuristic of Simon (1957) or the *take-the-best* heuristic of Gigerenzer and Goldstein (1996). It may be a worthwhile effort to perform a similar analysis like the one presented in this paper, but with alternative behavioural assumptions for the free agents.

Despite of its various reasonable alternatives, we want to stress that the search process modelled in this article has some intriguing features. Firstly, it is quite natural to assume that the free agents check for free tasks in order of their preferences. Secondly, the order in which free players propose

[12]For example, in stock exchange markets, humans trade simultaneously with computer programs. The computers act extremely fast, without any psychological biases, and they have superior computing power—hence they could be considered to be fully rational players.

to tasks, and the order in which they are rejected, does not influence the outcome assignment as the DAB procedure leads to a unique outcome. This follows from McVitie and Wilson (1971), as discussed in Section 3. Thirdly, in the deferred acceptance algorithm of Gale and Shapley (1962) there is no incentive for the proposing side, in our case the free agents, to misrepresent their preferences (cf. Dubins and Freedman (1981), Roth (1982)). In our context, this means that the free agents cannot improve their outcome by changing the order in which they approach tasks. So even if free agents would have enough information and computing power to act strategically, it would not be worthwhile to do this. In contrast, alternative models of search behaviour would have to take care of strategic manipulations on the free agents' parts. Of course, this makes handling our model *convenient*, yet does not support the empirical validity of the DAB assumption.

Other modifications to our model come to mind. It may be interesting to change the informational assumptions of the model. What if the productivities of the autonomous workers for different tasks is private knowledge of that worker?[13] Would there be a way to make the free agents reveal their private information? Could they even be incentivised to pick the task which would be best from the central planner's point of view? Designing a transfer scheme to achieve such goals would demand the free agents to be modelled with cardinal preferences. Arguably, this would reduce the robustness of the model, but it might lead to economically interesting dynamics similar to those which can be found in the famous labour market adjustment models of Crawford and Knoer (1981) and Kelso and Crawford (1982).

The idea of introducing autonomous agents in scenarios where the central planner normally has full control is not limited to assignment problems. Many other standard problems could be extended to include autonomous agents. Transportation or network flow with some transfers performed by autonomous agents, knapsack where autonomous agents are able to add their own items to the knapsack, and graph colouring with some nodes coloured by the agents are just a few examples.

References

Alarie, S., Audet, C., Jaumard, B., Savard, G., 2001. Concavity cuts for disjoint bilinear programming. Mathematical Programming 90, 373–398.

[13]We thank Ulrich Pferschy for suggesting this modification.

Audet, C., Hansen, P., Jaumard, B., Savard, G., 1999. A symmetrical linear maxmin approach to disjoint bilinear programming. Mathematical Programming 85, 573–592.

Benkler, Y., 2006. The Wealth of Networks: How Social Production Transforms Markets and Freedom. Yale University Press.

Bertsimas, D., Tsitsiklis, J., 1997. Introduction to linear optimization .

Brabham, D.C., 2008. Crowdsourcing as a Model for Problem Solving. Convergence 14, 75–90.

Crawford, V.P., Knoer, E.M., 1981. Job Matching with Heterogeneous Firms and Workers. Econometrica 49, 437–450.

Dubins, L.E., Freedman, D.A., 1981. Machiavelli and the Gale-Shapley Algorithm. American Mathematical Monthly 88, 485–494.

Gale, D., Shapley, L.S., 1962. College Admissions and the Stability of Marriage. American Mathematical Monthly 69, 9–15.

Gigerenzer, G., Goldstein, D.G., 1996. Reasoning the Fast and Frugal Way: Models of Bounded Rationality. Psychological Review 103, 650–669.

Howe, J., 2008. Crowdsourcing: Why the Power of the Crowd is driving the Future of Business. Crown Business.

Kelso, A.S., Crawford, V.P., 1982. Job Matching, Coalition Formation, and Gross Substitutes. Econometrica 50, 1483–1504.

Koopmans, T.C., Beckmann, M., 1957. Assignment Problems and the Location of Economic Activities. Econometrica 25, 53–76.

Luo, Z., Pang, J., Ralph, D., 1996. Mathematical programs with equilibrium constraints. Cambridge University Press.

Marcotte, P., Savard, G., 2005. Bilevel programming: A combinatorial perspective. Graph theory and combinatorial optimization , 191–217.

McVitie, D.G., Wilson, L.B., 1971. The Stable Marriage Problem. Communications of the ACM 14, 486–492.

Nikolic, I.A., Maikisch, H., 2006. Public-Private Partnerships and Collaboration in the Health Sector. HNP Discussion Paper (`http://info.worldbank.org/etools/docs/library/240103/PUBLIC~2.PDF`).

Riis, J.A., 2007. Hero Tales of the Far North. Echo Library.

Roth, A., Rothblum, U., Vande Vate, J., 1993. Stable matchings, optimal assignments, and linear programming. Math. Oper. Res. 18, 803–828.

Roth, A.E., 1982. The Economics of Matching: Stability and Incentives. Mathematics of Operations Research 7, 617–628.

Roth, A.E., Sotomayor, M., 1990. Two-Sided Matching - A Study in Game Theoretic Modeling and Analysis. Cambridge University Press.

Russell, S., Norvig, P., 2003. Artificial Intelligence: A Modern Approach. 2nd edition ed., Prentice-Hall, Englewood Cliffs, NJ.

Simon, H.A., 1957. Models of Man: Social and Rational. Wiley.

Templeton, S.K., 2007. Dentists refuse to treat bad Teeth. The Sunday Times, May 27.

Thieu, T., 1988. A note on the solution of bilinear programming problems by reduction to concave minimization. Mathematical Programming 41, 249–260.

Vande Vate, J., 1989. Linear programming brings marital bliss. Operations Research Letters 8, 147–153.

White, D., 1992. A linear programming approach to solving bilinear programmes. Mathematical Programming 56, 45–50.