Popular Matchings in the Capacitated House Allocation Problem

David F. Manlove* and Colin T.S. Sng

Department of Computing Science, University of Glasgow, Glasgow G12 8QQ, UK.
{davidm,sngts}@dcs.gla.ac.uk.

Abstract. We consider the problem of finding a popular matching in the *Capacitated House Allocation problem* (CHA). An instance of CHA involves a set of agents and a set of houses. Each agent has a preference list in which a subset of houses are ranked in strict order, and each house may be matched to a number of agents that must not exceed its capacity. A matching M is *popular* if there is no other matching M' such that the number of agents who prefer their allocation in M' to that in M exceeds the number of agents who prefer their allocation in M to that in M'. Here, we give an $O(\sqrt{C}n_1 + m)$ algorithm to determine if an instance of CHA admits a popular matching, and if so, to find a largest such matching, where C is the total capacity of the houses, n_1 is the number of agents and m is the total length of the agents' preference lists. For the case where preference lists may contain ties, we give an $O((\sqrt{C} + n_1)m)$ algorithm for the analogous problem.

1 Introduction

An instance I of the *Capacitated House Allocation problem* (CHA) comprises a bipartite graph $G = (A, H, E)$, where $A = \{a_1, a_2, ..., a_{n_1}\}$ is the set of *agents*, $H = \{h_1, h_2, ..., h_{n_2}\}$ is the set of *houses* and E is the set of edges in G. We let $n = n_1 + n_2$ and $m = |E|$. Each agent $a_i \in A$ ranks in strict order a subset of the set of houses (the *acceptable* houses for a_i) represented by his/her *preference list*. Each house $h_j \in H$ has a *capacity* $c_j \geq 1$ which indicates the maximum number of agents that may be matched to it. We assume that $m \geq \max\{n_1, n_2\}$, i.e. no agent has an empty preference list and each house is acceptable to at least one agent. We also assume that $c_j \leq n_1$ for each $h_j \in H$. Let $C = \sum_{j=1}^{n_2} c_j$ denote the sum of the capacities of the houses.

A *matching* M in I is a subset of E such that (i) each agent is matched to at most one house in M, and (ii) each house $h_j \in H$ is matched to at most c_j agents in M. If an agent $a_i \in A$ is matched in M, we denote by $M(a_i)$ the house that a_i is matched to in M. We define $M(h_j)$ to be the set of agents matched to h_j in M (thus $M(h_j)$ could be empty). Given two matchings M and M' in I, we say that an agent a_i *prefers* M' to M if either (i) a_i is matched in M' and unmatched in M, or (ii) a_i is matched in both M' and M and prefers $M'(a_i)$ to

* Supported by EPSRC grant GR/R84597/01 and RSE/Scottish Executive Personal Research Fellowship.

(a)
a_1: h_1 h_2 h_3
a_2: h_1 h_2 h_3
a_3: h_1 h_2 h_3

(b)
a_1: h_1 h_2
a_2: h_1

Fig. 1. Two instances of HA.

$M(a_i)$. Let $P(M', M)$ denote the set of agents who prefer M' to M. Then, M' is *more popular than* M if $|P(M', M)| > |P(M, M')|$, i.e. the number of agents who prefer M' to M is greater than the number of agents who prefer M to M'. Furthermore, a matching M in I is *popular* if there is no other matching M' in I that is more popular than M.

CHA is an example of a bipartite matching problem with one-sided preferences [1, 2, 7, 3]. These problems have applications in areas such as campus housing allocation in US universities [1], hence the problem name; in assigning probationary teachers to their first posts in Scotland; and in Amazon's DVD rental service. A variety of optimality criteria have been defined for such problems. Gärdenfors [6] first introduced the notion of a popular matching (also known as a *majority assignment*) in the context of voting theory. Alternatively, *Pareto optimality* [1, 2] is often regarded by economists as a fundamental property to be satisfied. A matching M is *Pareto optimal* if there is no matching M' such that some agent prefers M' to M, and no agent prefers M to M'. Finally, a matching is *rank maximal* [7] if it assigns the maximum number of agents to their first-choice houses, and subject to this, the maximum number of agents to their second-choice houses, and so on. However, Pareto optimal matchings and rank maximal matchings need not be popular.

Popular matchings were considered by Abraham et al. [3] in the context of the *House Allocation problem* (HA) – the special case of CHA in which each house has capacity 1. They gave an instance of HA in which no popular matching exists (see Figure 1(a)) and also noted that popular matchings can have different sizes (see Figure 1(b); in this HA instance the matchings $M_1 = \{(a_1, h_1)\}$ and $M_2 = \{(a_1, h_2), (a_2, h_1)\}$ are both popular). Abraham et al. [3] described an $O(n+m)$ algorithm for finding a maximum cardinality popular matching (henceforth a maximum popular matching) if one exists, given an instance of HA. They also described an $O(\sqrt{n}m)$ counterpart for the *House Allocation problem with Ties* (HAT) – the generalisation of HA in which agents' preferences may include ties.

Several other recent papers have also focused on popular matchings. Mahdian [8] gave some probabilistic results with respect to the existence of popular matchings in a random instance of HA. Abraham and Kavitha [4] considered popular matchings in a dynamic matching market in which agents and houses can enter and leave the market, and showed that there exists a 2-step *voting path* to compute a new popular matching from some initial matching after every such change, provided some popular matching exists. Also Mestre [10] studied a generalisation of the problem in which agents have a weight indicating their priority, and the objective is to compute a *weighted popular matching* M (i.e. there is no other matching M' such that the weighted majority of the agents prefer M' to M.)

In this paper, we consider popular matchings in instances of CHA and CHAT, where CHAT denotes the *Capacitated House Allocation problem with Ties* – the generalisation of CHA in which agents' preference lists may contain ties. Both CHA and CHAT are natural generalisations of the one-one HA and HAT models considered in [3] to the case where houses may have non-unitary capacity. We extend the characterisations and algorithms for popular matchings from [3] to these many-one settings. In particular, in Section 2, we develop a characterisation of popular matchings in a CHA instance I, and then use it to construct an $O(\sqrt{C}n_1 + m)$ algorithm for finding a maximum popular matching in I if one exists. In Section 3, we build a new characterisation of popular matchings in a CHAT instance I, and then use it to construct an $O((\sqrt{C}+n_1)m)$ algorithm for finding a maximum popular matching in I if one exists.

We finally remark that a straightforward solution to each of the problems of finding a maximum popular matching, given an instance of CHA or CHAT, may be to use "cloning". Informally, this entails creating c_j clones for each house h_j, to obtain an instance $C(I)$ of HAT (i.e. each house has capacity 1), and then applying the HAT algorithm of [3] to $C(I)$. However, we will show in Sections 2 and 3 that this method in general leads to slower algorithms than the direct approach that we will be using in each case.

2 Popular matchings in CHA

Characterising popular matchings. Let I be an instance of CHA. For each agent $a_i \in A$, let $f(a_i)$ denote the first-ranked house on a_i's preference list. Any such house h_j is called an *f-house*. For each $h_j \in H$, let $f(h_j) = \{a_i \in A : f(a_i) = h_j\}$ and $f_j = |f(h_j)|$ (possibly $f_j = 0$). Now let M be a matching in I. We say that a house $h_j \in H$ is *full* if $|M(h_j)| = c_j$, and *undersubscribed* if $|M(h_j)| < c_j$. We also create a unique *last resort* house $l(a_i)$ with capacity 1 for each agent $a_i \in A$, and append $l(a_i)$ to a_i's preference list. The following lemma is a vital first step in characterising popular matchings in I.

Lemma 1. *Let M be a popular matching in I. Then for every f-house h_j, $|M(h_j) \cap f(h_j)| = \min\{c_j, f_j\}$.*

Proof. We consider the following two cases.

– *Case (i):* Suppose $f_j \leq c_j$. We will show that $f(h_j) \subseteq M(h_j)$. For, suppose not. Then choose any $a_r \in f(h_j) \backslash M(h_j)$. We consider the subcases that (a) h_j is undersubscribed and (b) h_j is full. In subcase (a), promote a_r to h_j to obtain a more popular matching than M. In subcase (b), choose any $a_s \in M(h_j) \backslash f(h_j)$. Let $h_k = f(a_s)$. Then $h_k \neq h_j$. If h_k is undersubscribed, promote a_r to h_j and promote a_s to h_k to obtain a more popular matching than M. Otherwise, choose any $a_t \in M(h_k)$. We then promote a_r to h_j, promote a_s to h_k and demote a_t to $l(a_t)$ to obtain a more popular matching than M.

– *Case (ii):* Suppose $f_j > c_j$. If h_j is undersubscribed, then $f(h_j) \not\subseteq M(h_j)$ so there exists some $a_r \in f(h_j) \backslash M(h_j)$ that we can promote to h_j to obtain a more popular matching as in Case (i)(a). Hence, h_j is full. Now, suppose for a

contradiction that $M(h_j) \not\subseteq f(h_j)$. Then there exists some $a_s \in M(h_j) \backslash f(h_j)$. As $f_j > c_j$, it follows that $f(h_j) \not\subseteq M(h_j)$ so there exists some $a_r \in f(h_j) \backslash M(h_j)$. The remainder of the argument follows Case (i)(b).

Hence the following properties hold for the new matching. If $f_j \leq c_j$, then $f(h_j) \subseteq M(h_j)$. Otherwise, $M(h_j) \subseteq f(h_j)$ and $|M(h_j)| = c_j$. Thus, the condition in the statement of the lemma is now satisfied. □

For each agent a_i, we next define $s(a_i)$ to be the most-preferred house h_j on a_i's preference list such that either (i) h_j is a non-f-house, or (ii) h_j is an f-house such that $h_j \neq f(a_i)$ and $f_j < c_j$. Note that $s(a_i)$ must exist in view of $l(a_i)$. We refer to such a house h_j as an *s-house*. We remark that the set of f-houses need not be disjoint from the set of s-houses. It may be shown that a popular matching M will only match an agent a_i to either $f(a_i)$ or $s(a_i)$, as indicated by the next two lemmas (see [9] for the proofs).

Lemma 2. *Let M be a popular matching in I. Then no agent $a_i \in A$ can be matched in M to a house between $f(a_i)$ and $s(a_i)$ on a_i's preference list.*

Lemma 3. *Let M be a popular matching in I. Then no agent $a_i \in A$ can be matched in M to a house worse than $s(a_i)$ on a_i's preference list.*

Let $G = (A, H, E)$ be the underlying graph of I. We form a subgraph G' of G by letting G' contain only two edges for each agent a_i, that is, one to $f(a_i)$ and the other to $s(a_i)$. We say that a matching M is *agent-complete* in a given graph if it matches all agents in the graph. Clearly, in view of last resort houses, all popular matchings must be agent-complete in G'. However, G' need not admit an agent-complete matching if $s(a_i) \neq l(a_i)$ for some agent a_i. In conjunction with Lemmas 1-3, the graph G' gives rise to the following characterisation of popular matchings in I.

Theorem 1. *A matching M is popular in I if and only if*

1. *for every f-house h_j,*
   (a) *if $f_j \leq c_j$, then $f(h_j) \subseteq M(h_j)$;*
   (b) *if $f_j > c_j$, then $|M(h_j)| = c_j$ and $M(h_j) \subseteq f(h_j)$.*
2. *M is an agent-complete matching in the reduced graph G'.*

Proof. By Lemmas 1-3, any popular matching necessarily satisfies Conditions 1 and 2. We now show that these conditions are sufficient.

Let M by any matching satisfying Conditions 1 and 2 and suppose for a contradiction that M' is a matching that is more popular than M. Let a_i be any agent that prefers M' to M and let $h_k = M'(a_i)$. Since M is an agent-complete matching in G', and since G' contains only edges from a_i to $f(a_i)$ and $s(a_i)$, then $M(a_i) = s(a_i)$. Hence either (i) $h_k = f(a_i)$ or (ii) h_k is an f-house such that $h_k \neq f(a_i)$ and $f_k \geq c_k$, by definition of $s(a_i)$.

In Case (i), if $f_k < c_k$ then by Condition 1(a), $a_i \in M(h_k)$, a contradiction. Hence in both Cases (i) and (ii), $f_k \geq c_k$. In each of the cases that $f_k = c_k$ and $f_k > c_k$, it follows by Conditions 1(a) and 1(b) that $|M(h_k)| = c_k$ and $M(h_k) \subseteq$

```
1.   M := ∅;
2.   for each f-house h_j
3.       c'_j := c_j;
4.       if f_j ≤ c_j
5.           for each a_i ∈ f(h_j)
6.               M := M ∪ {(a_i, h_j)};
7.               delete a_i and its incident edges from G';
8.           c'_j := c_j − f_j;
9.    remove all isolated and full houses, and their incident edges, from G';
10.   compute a maximum matching M' in G' using capacities c'_j;
11.   if M' is not agent-complete in G'
12.       output "no popular matching exists"
13.   else
14.       M := M ∪ M';
15.       for each a_i ∈ A
16.           h_j := f(a_i);
17.           if f_j > c_j and |M(h_j)| < c_j and h_j ≠ M(a_i)
18.               promote a_i from M(a_i) to h_j in M;
```

Fig. 2. Algorithm Popular-CHA for finding a popular matching in CHA.

$f(h_k)$. Since h_k is full in M, it follows that $|M(h_k)\backslash M'(h_k)| \geq |M'(h_k)\backslash M(h_k)|$. Hence for every a_i who prefers $M'(a_i) = h_k$ to $M(a_i)$, there is a unique $a_j \in M(h_k)\backslash M'(h_k)$. But as $a_j \in M(h_k)$, it follows that $h_k = f(a_j)$. Hence a_j prefers $M(a_j)$ to $M'(a_j)$. Therefore, M is popular in I. □

Finding a popular matching. Theorem 1 leads to Algorithm Popular-CHA for finding a popular matching in a CHA instance I, or reporting that none exists, as shown in Figure 2. The algorithm begins by using a pre-processing step (lines 2-9) on G' that matches agents to their first-choice house h_j whenever $f_j \leq c_j$, so as to satisfy Condition 1(a) of Theorem 1.

Our next step computes a maximum cardinality matching M' (henceforth a maximum matching) in G', according to the adjusted house capacities c'_j that are defined following pre-processing. The subgraph G' can be viewed as an instance of the Upper Degree-Constrained Subgraph problem (UDCS) [5]. (An instance of UDCS is essentially the same as an instance of CHA, except that agents have no explicit preferences in the UDCS case; the definition of a matching is unchanged.) We use Gabow's algorithm [5] to compute M' in G' and then test whether M' is agent-complete. The pre-allocations are then added to M' to give M. As a last step, we ensure that M also meets Condition 1(b) of Theorem 1. For, suppose that $h_j \in H$ is an f-house such that $f_j > c_j$. Then by definition, h_j cannot be an s-house. Thus if $a_k \in M(h_j)$ prior to the third for loop, it follows that $a_k \in f(h_j)$. At this stage, if h_j is undersubscribed in M, we repeatedly promote any agent $a_i \in f(h_j)\backslash M(h_j)$ from $M(a_i)$ (note that $M(a_i)$ must be $s(a_i)$ and hence cannot be an f-house h_l such that $f_l > c_l$) to h_j until h_j is full, ensuring that $M(h_j) \subseteq f(h_j)$.

It is clear that the reduced graph G' of G can be constructed in $O(m)$ time.

The graph G' has $O(n_1)$ edges since each agent has degree 2 in G'. Clearly each of the pre- and post-processing steps involving the three for loop phases takes $O(n_1 + n_2)$ time. The complexity of Gabow's algorithm [5] for computing M' in G' is $O(\sqrt{C}n_1)$. Hence we obtain the following result concerning the complexity of Algorithm Popular-CHA.

Lemma 4. *Given an instance of CHA, we can find a popular matching, or determine that none exists, in $O(\sqrt{C}n_1 + m)$ time.*

It remains to consider the problem of finding a maximum popular matching in I. We begin by dividing the set of all agents into disjoint sets. Let A_1 be the set of all agents a_i with $s(a_i) = l(a_i)$, and let $A_2 = A - A_1$. We aim to find a matching M that satisfies the conditions of Theorem 1, and that minimises the number of A_1-agents who are matched to their last resort house.

We begin by constructing G', and carrying out the pre-processing step in lines 2-9 of Algorithm Popular-CHA on all agents in $A_1 \cup A_2$. We then try to find a maximum matching M' in G' that only involves the A_2-agents that remain after pre-processing and their incident edges. If M' is not an agent-complete matching of the agents in A_2 that remain after pre-processing, then G admits no popular matching by Theorem 1. Otherwise, we remove all edges in G' that are incident to a last resort house, and try to match A_1-agents to their first-choice houses. At each step, we try to match an additional A_1-agent to his/her first-choice house by finding an augmenting path with respect to M' using Gabow's algorithm for UDCS [5], so that we have a maximum matching of agents in $A_1 \cup A_2$ in G' at the end of this process. If any A_1-agent remains unmatched, we simply assign him/her to his/her last resort house, to obtain an agent-complete matching in G'. We also ensure that Condition 1(b) of Theorem 1 is met by executing the third for loop in Algorithm Popular-CHA. Clearly then, the matching so obtained, together with the pre-assignments from earlier, is a maximum popular matching, giving the following theorem.

Theorem 2. *Given an instance of CHA, we can find a maximum popular matching, or determine that none exists, in $O(\sqrt{C}n_1 + m)$ time.*

An alternative approach to our algorithm would be to use cloning. Given an instance I of CHA, we may obtain an instance J of HAT by creating c_j clones $h_j^1, h_j^2, ..., h_j^{c_j}$ of each house h_j in I, where each clone has a capacity of 1. In addition, we replace each occurrence of h_j in a given agent's preference list with the sequence $h_j^1, h_j^2, ..., h_j^{c_j}$, the elements of which are listed in a single tie at the point where h_j appears. We can then apply the $O(\sqrt{n}m)$ algorithm for HAT given by [3] to J in order to find a maximum popular matching in I.

We now compare the worst-case complexity of the above cloning approach with that of our algorithm. The underlying graph G_J of J contains $n' = n_1 + C$ nodes. Let $c_{min} = \min\{c_j : h_j \in H\}$, and for $a_i \in A$, let A_i denote the set of acceptable houses for a_i. Then the number of edges in G_J is $m' = \sum_{a_i \in A} \sum_{h_j \in A_i} c_j \geq mc_{min}$. Hence the complexity of applying the algorithm given by [3] to J is $\Omega(\sqrt{C}mc_{min})$. Recall that the complexity of Algorithm

Popular-CHA is $O(\sqrt{C}n_1+m)$. It follows that the cloning method is slower by a factor of $\Omega(\sqrt{C}c_{min})$ or $\Omega(mc_{min}/n_1)$ (note that $m \geq n_1$ and $c_{min} \geq 1$) according as $\sqrt{C}n_1 \leq m$ or $\sqrt{C}n_1 > m$ respectively. In the case that $c_{min} = \Omega(n_1)$, our approach offers an improvement by a factor of $\Omega(n_1^{3/2}n_2^{1/2})$ or $\Omega(m)$ respectively.

3 Popular matchings in CHAT

In this section, we generalise the characterisation of popular matchings together with Algorithm Popular-CHA as given in the previous section to the case that I is an instance of CHAT.

Characterising popular matchings. Let M be a popular matching in I. For each agent $a_i \in A$, let $f(a_i)$ denote the set of first-ranked houses on a_i's preference list (clearly it is possible that $|f(a_i)| > 1$ in view of ties in the preference lists). We refer to all such houses h_j as *f-houses* and we let $f(h_j) = \{a_i \in A : h_j \in f(a_i)\}$. Let $G = (A, H, E)$ be the underlying graph of I. Define $E_1 = \{(a_i, h_j) : a_i \in A \wedge h_j \in f(a_i)\}$ to be the set of *first-choice edges*. We define the *first-choice graph* of G as $G_1 = (A, H, E_1)$. For instances with strict preference lists, Lemma 1 implies that $M \cap E_1$ is a maximum matching in G_1. As the next lemma indicates (see [9] for the proof), this latter condition also extends to the CHAT case.

Lemma 5. *Let M be a popular matching in I. Then $M \cap E_1$ is a maximum matching in G_1.*

As Lemma 1 no longer holds in general in a CHAT instance, we work towards a new definition of s-houses by using some concepts from the theory of bipartite matching. Let M be a maximum matching in some bipartite graph G where all nodes have capacity 1. According to the Edmonds-Gallai Decomposition (see [11]), then the nodes of G can be partitioned into three disjoint sets: $\mathcal{E}$, $\mathcal{O}$ and $\mathcal{U}$. Nodes in $\mathcal{E}$, $\mathcal{O}$ and $\mathcal{U}$ are called *even, odd,* and *unreachable* respectively. A node v is even (odd) if there exists an alternating path of even (odd) length from an unmatched node in G to v. If no such alternating path exists, v is unreachable. Some fundamental properties of this node labelling (henceforth referred to as the *EOU labelling*) in relation to a maximum matching in G are summarised in Lemma 3.2 of [3].

Our aim is to obtain an EOU labelling of G_1 relative to a maximum matching M_1 of G_1 (as obtained by Gabow's algorithm [5], for example). However Lemma 3.2 of [3] applies directly only to the case where each node in the given bipartite graph has capacity 1. We obtain an EOU labelling of nodes in G_1 by a cloning process, as follows. The *cloned graph* $C(G_1)$ can be constructed from G_1 by replacing every house $h_j \in H$ with the clones $h_j^1, h_j^2, \ldots, h_j^{c_j}$. We then divide the capacity of each house among its clones by allowing each clone to have capacity 1. In addition, if $(a_i, h_j) \in G_1$, then we add $(a_i, h_j^k) \in C(G_1)$ for all k $(1 \leq k \leq c_j)$. We then adapt the maximum matching M_1 in G_1 to obtain a matching $C(M_1)$ in $C(G_1)$, as follows. If a house h_j in G_1 is matched to x_j agents $a_{i_1}, \ldots a_{i_{x_j}}$ in

M_1, then we add (a_{i_k}, h_j^k) to $C(M_1)$ for $1 \leq k \leq x_j$, so that $|C(M_1)| = |M_1|$ and $C(M_1)$ is a maximum matching in $C(G_1)$.

We next use $C(M_1)$ and $C(G_1)$ to obtain an EOU labelling of the nodes in $C(G_1)$, and hence G_1. Clearly, such a labelling in $C(G_1)$ is useful only if it can give a well-defined characterisation of EOU labels in G_1. Crucial to this is the need for the clones corresponding to each house $h_j \in H$ to have the same EOU label in $C(G_1)$, as stated by the next lemma (see [9] for the proof).

Lemma 6. *Let G_1 be the first-choice graph in I and let M_1 be a maximum matching in G_1. Define the cloned graph $C(G_1)$ and its corresponding maximum matching $C(M_1)$ as above. Then, given any house $h_j \in H$, any two clones of h_j in $C(G_1)$ have the same EOU label.*

We now use Lemma 6 to obtain an EOU labelling of the nodes in G_1. Clearly, in view of Lemma 6, a well-defined EOU labelling of $h_j \in H$ can be obtained by letting h_j inherit its EOU label from those of its clones. That is, we say that h_j is even, odd or unreachable in G_1 if its clones are even, odd or unreachable in $C(G_1)$ respectively. It is immediate that each agent can inherit its EOU label in G_1 from its corresponding label in $C(G_1)$. The next result is a consequence of Lemma 6 (see [9] for the proof).

Lemma 7. *Let M be a popular matching in I. Then every odd or unreachable house $h_j \in H$ satisfies $|M(h_j)| = c_j$ and $M(h_j) \subseteq f(h_j)$.*

Lemmas 6 and 7 give us the following analogue of Lemma 3.2 from [3] for CHAT.

Lemma 8. *Let G_1 be the first-choice graph in I and let M_1 be a maximum matching in G_1. Define $\mathcal{E}$, $\mathcal{O}$ and $\mathcal{U}$ to be the node sets corresponding to even, odd and unreachable nodes in an EOU labelling of G_1 with respect to M_1. Then:*

(a) The sets $\mathcal{E}$, $\mathcal{O}$ and $\mathcal{U}$ are pairwise disjoint. Every maximum matching in G_1 partitions the nodes into the same sets of even, odd and unreachable nodes.
(b) Every maximum matching M in G_1 satisfies the following properties:
  (i) every odd agent is matched to an even house in M;
  (ii) every odd house is full in M and matched only to even agents in M;
  (iii) every unreachable agent is matched to an unreachable house in M;
  (iv) every unreachable house is full in M and matched only to unreachable agents in M;
  (v) $|M| = |\mathcal{O}_A| + |\mathcal{U}_A| + \sum_{h_j \in \mathcal{O}_H} c_j$, where $\mathcal{U}_A$ is the set of unreachable agents, $\mathcal{O}_A$ is the set of odd agents and $\mathcal{O}_H$ is the set of odd houses.
(c) No maximum matching in G_1 contains an edge between two nodes in $\mathcal{O}$ or a node in $\mathcal{O}$ with a node in $\mathcal{U}$. There is no edge in G_1 connecting a node in $\mathcal{E}$ with a node in $\mathcal{U}$, or between two nodes of $\mathcal{E}$.

We are now in a position to define $s(a_i)$, the set of houses such that, in a popular matching M, if $a_i \in A$ is matched in M and $M(a_i) \notin f(a_i)$, then $M(a_i) \in s(a_i)$. We will ensure that any odd or unreachable house h_j is not a member of $s(a_i)$, since $|M(h_j)| = c_j$ and $M(h_j) \subseteq f(h_j)$ by Lemma 7. Hence,

we define $s(a_i)$ to be the set of highest-ranking houses in a_i's preference list that are even in G_1. Any such house is called an *s-house*. Clearly, it is possible that $|s(a_i)| > 1$, however, a_i is indifferent between all houses in $s(a_i)$. Furthermore, $s(a_i) \neq \emptyset$ due to the existence of last resort houses which are of degree 0 in G_1 (and thus even). However, $f(a_i)$ and $s(a_i)$ need not be disjoint. It turns out that Lemmas 2 and 3 also extend to CHAT as established by the following lemmas (see [9] for the proofs).

Lemma 9. *Let M be a popular matching in I. Then no agent $a_i \in A$ can be matched in M to a house between $f(a_i)$ and $s(a_i)$ on a_i's preference list.*

Lemma 10. *Let M be a popular matching in I. Then no agent $a_i \in A$ can be matched in M to a house worse than $s(a_i)$ on a_i's preference list.*

As was the case with CHA, we can also define a subgraph G' for the CHAT instance I by this time letting G' contain only edges from each agent a_i to houses in $f(a_i) \cup s(a_i)$. Clearly, all popular matchings must be agent-complete in G' in view of last resort houses. However, an agent-complete matching need not exist if $s(a_i) \neq \{l(a_i)\}$ for some agent a_i. Lemmas 5, 9 and 10 give rise to the following characterisation of popular matchings in I.

Theorem 3. *A matching M is popular in I if and only if*

1. *$M \cap E_1$ is a maximum matching in G_1, and*
2. *M is an agent-complete matching in the subgraph G'.*

Proof. By Lemmas 5, 9 and 10, any popular matching necessarily satisfies Conditions 1 and 2. We now show that these conditions are sufficient.

Let M be any matching satisfying Conditions 1 and 2. Suppose for a contradiction that M' is a matching that is more popular than M. Let a_i be any agent that prefers M' to M. Since a_i prefers $M'(a_i)$ to $M(a_i)$, M is an agent-complete matching in G', and G' only contains edges from a_i to $f(a_i) \cup s(a_i)$, it follows that $M(a_i) \in s(a_i)$, and $f(a_i)$ and $s(a_i)$ are disjoint. Hence, $M'(a_i)$ must be an odd or unreachable house in G_1, as $M(a_i)$ is the highest-ranked even house in a_i's preference list.

Let $h_{j_1} = M'(a_i)$. Since h_{j_1} is odd or unreachable, it follows by Condition 1 and Lemma 8(b) that $|M(h_{j_1})| = c_{j_1}$ and $M(h_{j_1}) \subseteq f(h_{j_1})$. Now since $a_i \in M'(h_{j_1}) \backslash M(h_{j_1})$, there exists a distinct agent $a_{k_1} \in M(h_{j_1}) \backslash M'(h_{j_1})$. If a_{k_1} is unmatched in M' or $M'(a_{k_1}) \notin f(a_{k_1})$, then a_{k_1} prefers M to M'. Otherwise, suppose $M'(a_{k_1}) \in f(a_{k_1})$. Let $h_{j_2} = M'(a_{k_1})$. Clearly, a_{k_1} is even or unreachable so that h_{j_2} must be odd or unreachable. It follows by Condition 1 and Lemma 8(b) that $|M(h_{j_2})| = c_{j_2}$ and $M(h_{j_2}) \subseteq f(h_{j_2})$. Hence, there exists an agent $a_{k_2} \neq a_{k_1}$ such that $a_{k_2} \in M(h_{j_2}) \backslash M'(h_{j_2})$ and $h_{j_2} \in f(a_{k_2})$. If a_{k_2} is unmatched in M' or $M'(a_{k_2}) \notin f(a_{k_2})$, then a_{k_2} prefers M to M'. Otherwise, suppose that $M'(a_{k_2}) \in f(a_{k_2})$. Let $h_{j_3} = M'(a_{k_2})$. Then there exists an agent $a_{k_3} \in M(h_{j_3}) \backslash M'(h_{j_3})$ by a similar argument for a_{k_2}. Note that possibly $h_{j_3} = h_{j_1}$, but we must be able to choose $a_{k_3} \neq a_{k_1}$, for otherwise $|M'(h_{j_1})| > |M(h_{j_1})|$, which is a contradiction since $|M(h_{j_1})| = c_{j_1}$. Thus, a_{k_3} is a distinct agent, so that

1. Build subgraph $G_1=(A, H, E_1)$, where $E_1=\{(a_i, h_j) : a_i \in A \wedge h_j \in f(a_i)\}$.
2. Compute a maximum matching M_1 of first-choice edges in G_1.
3. Obtain an EOU labelling of G_1 using $C(G_1)$ and $C(M_1)$.
4. Build subgraph $G'=(A, H, E')$, where $E'=\{(a_i, h_j) : a_i \in A \wedge h_j \in f(a_i) \cup s(a_i)\}$.
5. Delete all edges in G' connecting two odd nodes, or connecting an odd node with an unreachable node. (This step does not delete an edge of M_1.)
6. Find a maximum matching M in the reduced graph G' by augmenting M_1.
7. If M is not agent-complete in G', then output "No popular matching exists", otherwise return M as a popular matching in I.

Fig. 3. Algorithm Popular-CHAT for finding a popular matching in CHAT.

we can repeat the above argument to identify an alternating path P in which houses need not be distinct, but agents are distinct. Clearly, P must terminate at some agent a_{k_r} as the number of agents are finite. Furthermore, it must be the case that a_{k_r} is unmatched in M' or $M'(a_{k_r}) \notin f(a_{k_r})$ so that for every a_i that prefers M' to M, there must exist a distinct a_{k_r} that prefers M to M'.

Finally, we note the uniqueness of a_{k_r}. If there exists another agent a'_i who prefers M' to M, then we can build another alternating path – it is possible that some of the houses are those already used in previous alternating paths such as P. However, it must be the case (from our argument that a_{k_3} is a distinct agent) that we are always able to identify distinct agents not already used in previous alternating paths, as each house on the path is odd or unreachable, and thus full in M. Hence, M is popular in I. □

Finding a popular matching. Theorem 3 leads to Algorithm Popular-CHAT for finding a popular matching in I of CHAT or reporting that none exists, as shown in Figure 3. The next lemma is an important step in establishing the correctness of the algorithm.

Lemma 11. *Algorithm Popular-CHAT constructs a matching M such that $M \cap E_1$ is a maximum matching of G_1.*

Proof. (Sketch – see [9] for the full proof.) Firstly, we claim that only first-choice edges are incident to odd nodes and unreachable houses in G' at the end of Step 4, using our definition of s-houses and Lemma 8(b). Define a *second-choice* edge as belonging to the edge set $\{(a_i, h_j) \in E' : h_j \in s(a_i) \wedge s(a_i) \not\subseteq f(a_i)\}$. By the claim, and by Lemma 8(c), the only first-choice edges in G' after Step 5 are those between (i) odd agents and even houses, (ii) even agents and odd houses, and (iii) unreachable agents and unreachable houses; the only second-choice edges are those between (i) even agents and even houses, and (ii) unreachable agents and even houses. Moreover no edge of M_1 is deleted by Step 5 of the algorithm. It follows that in Step 6, odd agents must remained matched to their first-choice houses, and by an argument involving alternating paths, unreachable agents cannot become worse off. Only even agents may become worse off, so that at least $|\mathcal{O}_A|+|\mathcal{U}_A|+\sum_{h_j \in \mathcal{O}_H} c_j$ first-choice edges are matched in the matching M. It thus follows by Lemma 8(b) that $M \cap E_1$ is a maximum matching of G_1. □

Hence if Algorithm Popular-CHAT returns a matching M, then M is both an agent-complete matching in G' and $M \cap E_1$ is a maximum matching of G_1 by Lemma 11. Hence M is a popular matching in I by Theorem 3.

We now consider the complexity of Algorithm Popular-CHAT. Let F be the number of first-choice edges in G, and let $c_{max} = \max\{c_j : h_j \in H\}$; then $c_{max} \leq n_1$. Clearly G_1 can be constructed in $O(F + n_2)$ time. We use Gabow's algorithm [5] to compute a maximum matching M_1 in G_1 in $O(\sqrt{C}F)$ time. We next use $C(M_1)$ in $C(G_1)$ to compute an EOU labelling of G_1. The total number of edges in $C(G_1)$ is $O(c_{max}F)$. We first use a pre-processing step to label each unmatched agent and each undersubscribed house as even. Clearly, this step takes $O(n)$ time. Next, breadth-first search may be used on $C(G_1)$ to search for alternating paths with respect to $C(M_1)$, building up odd or even labels for every node encountered. This step labels all odd and even (matched) agents, and all odd and even (full) houses and takes $O(c_{max}F + n_2)$ time. Any remaining unlabelled nodes must be unreachable and we can directly label these nodes in G_1 in $O(n)$ time. Thus, the total time complexity of this step is $O(c_{max}F + n_2) = O(n_1F + n_2)$. The EOU labelling of G_1 is then used to construct G' and to delete certain edges from G' at Steps 4 and 5 of the algorithm, both of which take $O(m)$ time overall.

Finally, we use Gabow's algorithm again to obtain the maximum matching M in G' in $O(\sqrt{C}(F+S))$ time, where S is the number of second-choice edges in G'. The following result gives the overall run-time of Algorithm Popular-CHAT.

Lemma 12. *Given an instance of CHAT, we can find a popular matching, or determine that none exists, in $O((\sqrt{C} + n_1)m)$ time.*

It now remains to consider the problem of finding a maximum popular matching in I. The aim is to find a matching that satisfies the conditions of Theorem 3 and that minimises the number of agents who are matched to their last resort houses. We begin by firstly using Algorithm Popular-CHAT to compute a popular matching M in I, assuming such a matching exists. Then $M \cap E_1$ is a maximum matching in G_1. We remove all edges in G' (and thus from M) that are incident to a last resort house. Clearly, M still satisfies the property that $M \cap E_1$ is a maximum matching in G_1, but M need not be maximum in G' if agents become unmatched as a result of the edge removals. Thus, we obtain a new maximum matching M' from M by using Gabow's algorithm on G' again. If M' is not agent-complete in G', we simply assign any agent who remains unmatched in M' to their last resort house to obtain an agent-complete matching. Using an argument similar to that in the proof of Lemma 11, it follows that $M' \cap E_1$ is a maximum matching of G_1. Thus, M' is a maximum popular matching in I. Clearly the overall complexity of this approach is as for Algorithm Popular-CHAT, giving the following result.

Theorem 4. *Given an instance of CHAT, we can find a maximum popular matching, or report that no such matching exists, in $O((\sqrt{C} + n_1)m)$ time.*

We may compare the complexity of our direct approach for CHAT to that obtained using cloning on I together with the algorithm of [3] on the cloned

instance of I. As in Section 2, the latter approach takes $\Omega(\sqrt{C}m' + C)$ time, where $m' = \sum_{a_i \in A} \sum_{h_j \in A_i} c_j$. The complexity of Algorithm Popular-CHAT may be rewritten as $O(\sqrt{C}m + m_F + C)$, where $m_F = \sum_{a_i \in A} \sum_{h_j \in f(a_i)} c_j$. Clearly $m_F \leq m'$. Since $m' \geq mc_{min}$, the first term in the complexity function of the cloning method is slower than the first term in that of Algorithm Popular-CHAT by a factor of $\Omega(c_{min})$, which is $\Omega(n_1)$ if $c_j = \Omega(n_1)$ for each $h_j \in H$.

4 Concluding remarks

We conclude with the following open problem. Suppose that we are presented with an instance J of CHA or CHAT in which the houses have preferences over the agents. Real-life applications of such a problem exist in many centralised matching markets such as the National Resident Marketing Program (NRMP) [12] and counterpart schemes in Canada and Scotland. Then, what is the complexity of finding a maximum popular matching in J if one exists?

Acknowledgement

We would like to thank Rob Irving for helpful discussions concerning this paper.

References

1. A. Abdulkadiroğlu and T. Sönmez. Random serial dictatorship and the core from random endowments in house allocation problems. *Econometrica*, 66(3):689–701, 1998.
2. D.J. Abraham, K. Cechlárová, D.F. Manlove, and K. Mehlhorn. Pareto optimality in house allocation problems. In *Proceedings of ISAAC 2004: the 15th Annual International Symposium on Algorithms and Computation*, volume 3341 of *Lecture Notes in Computer Science*, pages 3–15. Springer, 2004.
3. D.J. Abraham, R.W. Irving, T. Kavitha, and K. Mehlhorn. Popular matchings. In *Proceedings of SODA '05: the 16th ACM-SIAM Symposium on Discrete Algorithms*, pages 424–432. ACM-SIAM, 2005.
4. D.J. Abraham and T. Kavitha. Dynamic matching markets and voting paths. In *Proceedings of SWAT 2006: the 10th Scandinavian Workshop on Algorithm Theory*, volume 4059 of *Lecture Notes in Computer Science*, pages 65–76. Springer, 2006.
5. H.N. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In *Proceedings of STOC '83: the 15th Annual ACM Symposium on Theory of Computing*, pages 448–456. ACM, 1983.
6. P. Gärdenfors. Match making: assignments based on bilateral preferences. *Behavioural Science*, 20:166–173, 1975.
7. R.W. Irving, T. Kavitha, K. Mehlhorn, D. Michail, and K. Paluch. Rank-maximal matchings. In *Proceedings of SODA '04: the 15th ACM-SIAM Symposium on Discrete Algorithms*, pages 68–75. ACM-SIAM, 2004.
8. M. Mahdian. Random popular matchings. In *Proceedings of EC '06: the 7th ACM Conference on Electronic Commerce*, pages 238–242. ACM, 2006.

9. D.F. Manlove and C.T.S. Sng. Popular matchings in the Capacitated House Allocation problem. Technical Report TR-2006-222, University of Glasgow, Department of Computing Science, 2006.
10. J. Mestre. Weighted popular matchings. In *Proceedings of ICALP '06: the 33rd International Colloquium on Automata, Languages and Programming*, volume 4051 of *Lecture Notes in Computer Science*, pages 715–726. Springer, 2006.
11. W.R. Pulleyblank. Matchings and extensions. In R.L. Graham, M. Grotschel, and L. Lovasz, editors, *Handbook of Combinatorics*, volume 1, chapter 3, pages 179–232. North-Holland, 1995.
12. A.E. Roth. The evolution of the labor market for medical interns and residents: a case study in game theory. *Journal of Political Economy*, 92(6):991–1016, 1984.