CREATING SPANNING TREES IN WAITER-CLIENT GAMES

GRZEGORZ ADAMSKI, SYLWIA ANTONIUK, MAŁGORZATA BEDNARSKA-BZDĘGA, DENNIS CLEMENS, FABIAN HAMANN, AND YANNICK MOGGE

Abstract. For a positive integer n and a tree T_n on n vertices, we consider an unbiased Waiter-Client game $\mathrm{WC}(n, T_n)$ played on the complete graph K_n, in which Waiter's goal is to force Client to build a copy of T_n. We prove that for every constant $c < 1/3$, if $\Delta(T_n) \leqslant cn$ and n is sufficiently large, then Waiter has a winning strategy in $\mathrm{WC}(n, T_n)$. On the other hand, we show that there exist a positive constant $c' < 1/2$ and a family of trees T_n with $\Delta(T_n) \leqslant c'n$ such that Client has a winning strategy in the $\mathrm{WC}(n, T_n)$ game for every n sufficiently large. We also consider the corresponding problem in the Client-Waiter version of the game.

1. Introduction

We consider finite, two-person, perfect-information games called unbiased Waiter-Client games, which are closely related to Maker-Breaker and Avoider-Enforcer positional games (cf. [2], [8]). Given a hypergraph $\mathcal{H} = (\mathcal{X}, \mathcal{F})$, in the Waiter-Client game played on board $\mathcal{H}$, the first player, called Waiter, offers the second player, called Client, two previously unclaimed elements of $\mathcal{X}$. Client then chooses one of these elements which he keeps and the other element goes to Waiter. If in the final round only one unclaimed element remains, then it goes to Waiter. The game ends when all elements of $\mathcal{X}$ have been claimed. Waiter wins if at the end there is a set $F \in \mathcal{F}$ with all its elements claimed by Client; otherwise Client is the winner.

The rules of the Client-Waiter game on $\mathcal{H}$ are similar, with the difference that if at the end of the game there is a set $F \in \mathcal{F}$ with all its elements claimed by Client, then Client is the winner. Also the last round rule in the Client-Waiter game is different: if only one unclaimed element remains, it goes to Client. Let us add that Waiter-Client and Client-Waiter games were introduced by Beck [1] under the names of Picker-Chooser and Chooser-Picker, respectively.

Usually, Waiter-Client (or Client-Waiter) games are studied in a graph setting, i.e. it is assumed that $\mathcal{X}$ is the set of edges of the complete graph K_n and Waiter tries to force Client to build (or Client wants to build) a graph having a given graph property (which is just a family of graphs). In other words, the family $\mathcal{F}$ consists of the edge-sets of a given graph family. With some abuse of notation, we say that such games are played on K_n.

There is an interesting relation between games played on K_n and random processes. For instance, the case where in a Waiter-Client game on K_n Waiter plays randomly is the well-known Achlioptas process (without replacement). Furthermore, a classic observation shows that sometimes the winner of the game could be predicted using a heuristic known as the *probabilistic intuition*. This intuition

The research of the fourth and sixth author is supported by Deutsche Forschungsgemeinschaft (Project CL 903/1-1).

suggests that the result of the game is related to the properties of the (binomial) random graph $G(n, 1/2)$, in which the number of edges is roughly the same as the number of Client's edges at the end of the game. A fascinating example of this phenomenon was observed by Beck [2] who proved that the size of the largest clique that Waiter can force in a Waiter-Client game played on K_n is $2\log_2 n - 2\log_2\log_2 n + O(1)$. The same holds for the Client-Waiter version of the problem, i.e. this is the size of the largest clique Client can build in K_n. Note that $2\log_2 n - 2\log_2\log_2 n + O(1)$ is also the clique number of the random graph $G(n, 1/2)$ with high probability, see e.g. [4]. We return to random intuition hints in the last part of this section.

In this paper, we focus on Waiter-Client games on K_n where Waiter tries to force Client to build a copy of a fixed spanning tree T_n. We denote such a game by $\mathrm{WC}(n, T_n)$. Throughout the paper we assume that every edge claimed by Client is colored red, while every edge in the Waiter's graph is colored blue. Note that if we relax the assumption that the spanning tree is fixed and we allow any spanning tree, then Waiter's task becomes much easier. Indeed, Csernenszky et al. [6] showed that Waiter can achieve such a goal playing on any graph G (not necessarily complete), if and only if G contains two edge-disjoint spanning trees. Hence, Waiter can force a spanning red tree in K_n for every $n \geqslant 4$.

Returning to $\mathrm{WC}(n, T_n)$, we are interested in the outcome of the game depending on the maximum degree of T_n and for large n. The last three authors and their co-authors proved in [5] that if n is large enough, Waiter can force a red Hamilton path within $n-1$ rounds. Furthermore, Waiter can force a red copy of any fixed tree T_n with $\Delta(T_n) \leqslant c\sqrt{n}$ within n rounds, for some suitable constant c. Though the authors of [5] focused on the problem of building a given tree fast, they also posed the following question regarding maximum degrees.

Question 1.1. *What is the largest integer $D(n)$ such that for every tree T_n with n vertices and the maximum degree at most $D(n)$ Waiter has a winning strategy in $WC(n, T_n)$?*

The above mentioned result in [5] implies that $D(n) = \Omega(\sqrt{n})$. On the other hand, it is known that $D(n) \leqslant n/2 + O(\sqrt{n \log n})$. Indeed, Waiter cannot even force a red star of size $n/2 + c\sqrt{n \log n}$ for a sufficiently large constant $c > 0$, which follows from a much more general results on discrepancy games played on hypergraphs, proved by Beck ([2], Theorem 18.3). Our main result is that $n/3 + o(n) \leqslant D(n) \leqslant (1/2 - c)n + o(n)$ for some positive constant c. We state it as the following two theorems.

Theorem 1.2. *For every $\varepsilon \in \left(0, \frac{1}{3}\right)$ there exist positive constants b and n_0 such that the following holds. Let T_n be a tree on $n \geqslant n_0$ vertices with $\Delta(T_n) < \left(\frac{1}{3} - \varepsilon\right) n$. Then Waiter has a winning strategy in $WC(n, T_n)$. Furthermore, she can obtain her goal within at most $n + b\sqrt{n}$ rounds.*

Theorem 1.3. *There are positive constants c and n_0 such that the following holds for every $n \geqslant n_0$. There exists a tree T_n with n vertices and $\Delta(T_n) < \left(\frac{1}{2} - c\right) n$ such that Client has a winning strategy in $WC(n, T_n)$.*

We prove the above theorem with $c = 0.001$ and make no further effort to optimize it. Though the improvement from $D(n) \leqslant 0.5n + o(n)$ to $D(n) \leqslant 0.499n + o(n)$ seems cosmetic, we think that it is important for predicting a proper constant C in the desired formula $D(n) = Cn + o(n)$, provided such a constant C exists. The linear bounds on $D(n)$ prove that $D(n)$ is far from $\Theta(n/\log(n))$ suggested by the random graph $G(n, 1/2)$ behavior [11]. Nonetheless, our argument for Theorem 1.3 involves randomness. Namely, we describe a randomized strategy for Client that allows avoiding a red copy of a given tree with high probability.

Finally, let us mention the Client-Waiter version of the above game. In contrast to Waiter-Client games, here the random graph intuition may help. It is known [10] that there exists a tree T_n with n vertices and $\Delta(T_n) = O(n/\log(n))$ such that with high probability the random graph $G(n, 1/2)$ has no copy of T_n. We prove that a similar phenomenon occurs in Client-Waiter games.

Theorem 1.4. *There are positive constants c and n_0 such that the following holds. For every $n \geqslant n_0$ there exists a tree T_n with n vertices and $\Delta(T_n) \leqslant \frac{cn}{\log(n)}$ such that in a Client-Waiter game on K_n, Waiter can prevent Client from claiming a red copy of T_n.*

Organization of the paper. In Section 2 we collect useful probability tools, positional games theory tools and some results on embedding trees. In Section 3 we prove a few lemmas and theorems regarding forcing red forests in Waiter-Client games. Some of them may be of independent interest. For example we show (see Theorem 3.17) that if $n \geqslant 2$ and F is a forest with at most n vertices and exactly m edges, such that every of its components has less than $m/3$ edges, then Waiter can force a red copy of F in K_n.

Theorem 1.2 is proved in Section 4, while Theorem 1.3 is proved in Section 5. Section 6 contains the proof of Theorem 1.4. We add some concluding remarks at the end of the paper.

2. Preliminaries

2.1. **Notation.** First of all, we set $[n] := \{k \in \mathbb{N} : \ 1 \leqslant k \leqslant n\}$ for every positive integer n.

Most of our graph notation is standard and follows [12]. Let G be any graph. Then we write $V(G)$ and $E(G)$ for the vertex set and the edge set of G, respectively, and we set $v(G) := |V(G)|$ and $e(G) := |E(G)|$. If $\{v, w\}$ is a pair of vertices, we write vw for short. The neighborhood of a vertex v in graph G is $N_G(v) := \{w \in V(G) : vw \in E(G)\}$, and its degree is $\deg_G(v) := |N_G(v)|$. The maximum degree in G is $\Delta(G) := \max_{v \in V(G)} \deg_G(v)$. Given any $A, B \subseteq V(G)$ and $v \in V(G)$, we set $E_G(A) := \{vw \in E(G) : \ v, w \in A\}$, $e_G(A) := |E_G(A)|$, $E_G(A, B) := \{vw \in E(G) : \ v \in A, w \in B\}$, $e_G(A, B) := |E_G(A, B)|$, $\deg_G(v, A) := |N_G(v) \cap A|$, and $N_G(A) := (\bigcup_{v \in A} N_G(v)) \backslash A$. Note that whenever the graph G is clear from the context, we may omit the subscript G in all definitions above. Given $A \subset V(G)$, we let $G[A] = (A, E_G(A))$ be the subgraph of G induced by A, and we set $G - A := G[V(G) \backslash A]$. Similarly, if we have disjoint sets $A, B \subset V(G)$, we write $G[A, B] := (A \cup B, E_G(A, B))$. If v is a vertex in G, we shortly write $G - v$ for $G - \{v\}$. Given $B \subseteq E(G)$, we also let $G \backslash B := (V(G), E(G) \backslash B)$. The disjoint union of graphs G and G' is denoted by $G \cup G'$. Moreover, an embedding of a graph H into a graph G is an injective map $f : V(H) \to V(G)$ such that $vw \in E(H)$ implies $f(v)f(w) \in E(G)$.

We will consider forests F with roots, where we may allow that the components have up to two roots. We say that a component of a forest is rooted if it contains exactly one root, and we say that it is double-rooted if it has exactly two roots. By a rooted forest we mean a forest such that each of its components has exactly one root. A leaf of F is a vertex of degree 1 which is not a root (if F has no roots, then every vertex of degree 1 is a leaf). With $L(F)$ we denote the set of leaves in F. A bare path in F is a path in F such that all its inner vertices have degree 2 in F. A leaf matching in F is a matching such that each of its edges contains a vertex from $L(F)$.

Assume that some Waiter-Client game is in progress. We let W and C denote the graphs consisting of Waiter's (blue) edges and Client's (red) edges, respectively, with the vertex set being equal to the graph that the game is played on. If an edge belongs to $C \cup W$, we say that it is colored, or alternatively we say that it is claimed. Otherwise, we say that the edge is uncolored or free. A vertex is free if all edges incident to it are free. We say that Waiter can force a graph G withing t rounds if Waiter has a strategy such that after at most t rounds there is a red copy of G at the board.

In some of our arguments Waiter forces a red copy of a given forest F with roots in a greedy kind of way. That is, there is a sequence $F_1 \subset F_2 \subseteq \ldots \subseteq F_s = F$ with F_1 containing the roots of F, and for each i, Waiter makes sure that after a total of $e(F_i)$ rounds a red copy $\bar{F}_i$ of F_i is created. Implicitly, this way we obtain an embedding $f_i : V(F_i) \to V(\bar{F}_i)$ which extends the earlier embedding f_{i-1} and which fixes the images of the the vertices of F_i. Having such a strategy, we sometimes simply say that Waiter embeds the graph F_i into the given board. Moreover, if the image of a vertex v under f_i is a vertex x, then we simply say that Waiter maps v to x.

2.2. **Probabilistic tools.** In our probabilistic argument, we will use a variant of the Chernoff bound, for a sum of independent random variables Z_i such that the number of terms is a random variable, not necessarily independent of Z_i.

Lemma 2.1. *For $n \in \mathbb{N}$ and $\rho \in (0,1)$, let $(Z_i)_{i=1}^n$ be a sequence of i.i.d. random variables with probability distribution*

$$Z_i = \begin{cases} 1, & \textit{with probability } \rho, \\ 0, & \textit{with probability } 1-\rho. \end{cases}$$

Then for any random variable T taking values in the set $\{0, 1, \ldots, n\}$, we have

$$\mathbb{P}\left(\left|\sum_{i=1}^{T} Z_i - T\rho\right| > 2\sqrt{n \log n}\right) = o(n^{-2}).$$

Proof. Using the union bound and the Chernoff inequality (see e.g. [9]), we obtain

$$\mathbb{P}\left(\left|\sum_{i=1}^{T} Z_i - T\rho\right| > 2\sqrt{n \log n}\right) \leqslant \sum_{k=1}^{n} \mathbb{P}\left(\left|\sum_{i=1}^{k} Z_i - k\rho\right| > 2\sqrt{n \log n}\right) \leqslant$$

$$\leqslant \sum_{k=1}^{n} 2\exp\left(\frac{-8n\log n}{k}\right) \leqslant 2n\exp(-8\log n) = o(n^{-2}).$$

□

We say that an event, depending on n, holds asymptotically almost surely (a.a.s.) if it holds with probability tending to 1 if n tends to infinity.

2.3. **Positional games tools.** We will use a variant of the Erdős-Selfridge Breaker's winning criterion (see e.g. [2, 8]), adapted to the Client-Waiter version by the third author, which can be also applied for Waiter-Client games.

Theorem 2.2 (Corollary 1.4 in [3]). *Consider a Waiter-Client game on a hypergraph $(\mathcal{X}, \mathcal{F})$ satisfying*

$$\sum_{F \in \mathcal{F}} 2^{-|F|+1} < 1.$$

Then Waiter has a strategy to force Client to claim at least one element in each of the sets in $\mathcal{F}$.

3. Forcing forests

For the proof of Theorem 1.2 we aim to provide Waiter's strategy that forces Client to build any fixed n-vertex tree T whose maximum degree is at most $\left(\frac{1}{3} - \varepsilon\right) n$. For this, we will distinguish trees by certain structural properties, but in any case, the overall strategy for Waiter will then be to first force a red copy of a subforest $F \subseteq T$ which contains all vertices of large degree in T, and then she will complete this subforest to a copy of T while making use of the structural properties mentioned above. Note that in such a procedure, when F is already embedded, we also fix the images of some of the vertices in order to be able to embed the remaining forest $T \backslash E(F)$. This is why we need to study games in which Waiter wants to force rooted forests. Below we prepare the tools for our main strategy, which is given in Section 4, by collecting and proving several lemmas regarding forcing different kinds of forests and forests with roots. A rich collection of lemmas is used to prove Theorems 3.17 and 3.22 – only these two results, together with Lemma 3.18, are applied in the proof of Theorem 1.2.

3.1. **Forcing Hamilton paths and perfect matchings.** The following two lemmas are more quantitative versions of results from [5] on forcing Hamilton paths with fixed endpoints.

Lemma 3.1. *For $n \geqslant 5$, Waiter can force Client to build a red Hamilton path on K_n within $n-1$ rounds and such that one of its endpoints is incident with at most two blue edges.*

Proof. Let v be any vertex of K_n. In each of the first two rounds Waiter offers any two free edges incident with v. The result is a red path P on three vertices whose endpoints are not incident with any other colored edge outside the path. Next, in each of the following $n-3$ rounds, let P be the current red path, and let w be any vertex not in P. Then Waiter offers the two edges between w and the endpoints of P, hence extending the red path P by one vertex. By the end of round $n-1$, the path P is a Hamilton path. Let w be the vertex which was added in the last round. Then there are at most two blue edges incident with w: the blue edge that was offered in the last round, and maybe one more edge if vw was offered within the first two rounds. □

Lemma 3.2. *Let $x, y \in V(K_n)$ with $n \geqslant 8$. Then within n rounds Waiter can force Client to build a red Hamilton path between x and y in K_n.*

Proof. Using Lemma 3.1, within $n-3$ first rounds Waiter can force a red Hamilton path $P = (v_1, v_2, \ldots, v_{n-2})$ on $K_n - \{x, y\}$ in which v_1 is incident with at most two blue edges. In round $n-2$ she offers the edges xv_{n-2} and yv_{n-2}, and without loss of generality we can assume that Client claims $v_{n-2}y$. Let i, j, with $3 \leqslant i, j \leqslant n-2$, be such that v_1v_i and v_1v_j are free. This is possible since so far v_1 is incident with at most two blue edges. Then in round $n-1$ Waiter offers edges v_1v_i and v_1v_j, and again without loss of generality we can assume that Client claims the edge v_1v_i. Finally, in round n, Waiter offers the edges xv_1 and xv_{i-1}. It is easy to see that no matter which edge Client claims, a red Hamilton path is built as required. □

The next lemma shows that Waiter can force quickly a perfect matching in an almost complete bipartite graph (with equal bipartition).

Lemma 3.3. *Let $t \geqslant 4$ and $K_{t,t}^-$ be the graph obtained from the complete bipartite graph $K_{t,t}$ by removing one edge. By offering edges of $K_{t,t}^-$ only, Waiter can force a red copy of a perfect matching in $K_{t,t}^-$ within $t+1$ rounds.*

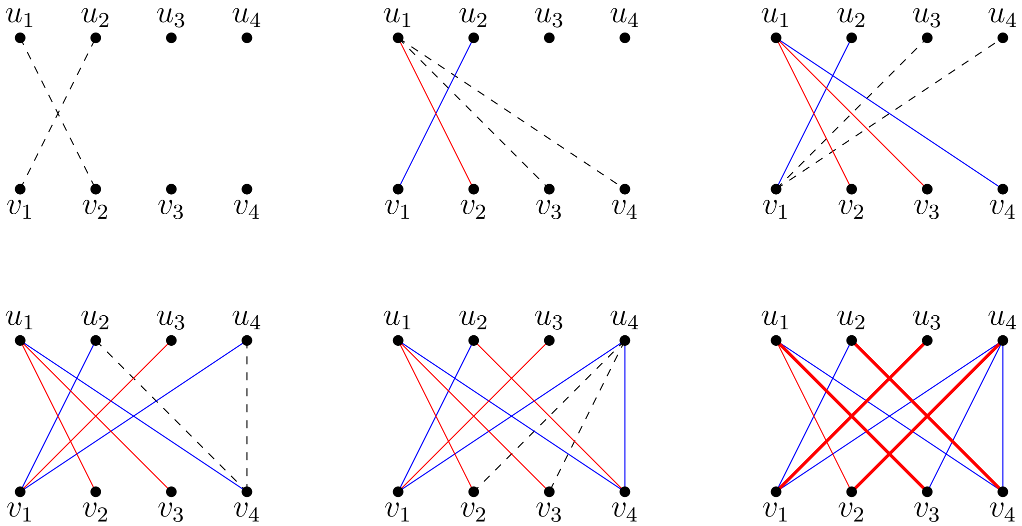

FIGURE 3.1. In order to build a perfect matching in $K_{4,4}^-$, in each round Waiter offers the two dashed edges.

Proof. We will use induction. First consider the case $t = 4$. Let $\{u_1, u_2, u_3, u_4\}$ and $\{v_1, v_2, v_3, v_4\}$ be the bipartition classes of $K_{4,4}^-$. Assume that u_1v_1 is the missing edge. Waiter plays in such a way that in each round Client's choices are irrelevant for the obtained structure of red and blue graphs, but they do have impact on the vertex labels in the edges offered by Waiter. (see Figure 3.1).

Waiter starts by offering the edges u_1v_2 and u_2v_1. By symmetry we can assume that u_1v_2 becomes red. Then Waiter offers the edges u_1v_3 and u_1v_4. Client chooses one of them, say u_1v_3. Then Waiter offers the edges u_3v_1 and u_4v_1. Again we can assume that Client colors for instance u_3v_1 red. Next Waiter offers the edges u_2v_4 and u_4v_4. Without loss of generality Client colors u_2v_4 red. Finally, Waiter offers the edges u_4v_2 and u_4v_3. Client colors any of them red, say u_4v_2, and we get the desired red matching, namely $\{u_1v_3, u_2v_4, u_3v_1, u_4v_2\}$, within 5 rounds.

Now for $t > 4$, suppose that uv is the missing edge. Then in the first round Waiter offers one edge incident with u, and the other incident with v, say uv' and vu'. By symmetry we can assume that Client colors uv' red. Then we remove from our graph the vertices u and v' and the blue edge vu', and the problem reduces to forcing a perfect matching in $K^-_{t-1,t-1}$ (now vu' is the missing edge), which by inductive assumption can be done within t rounds. Altogether, we get a perfect matching in $K^-_{t,t}$ within $t+1$ rounds. $\square$

3.2. **Rooted forest-tuples.** We begin by defining a couple of technical notions used extensively throughout the remaining part of this section.

Definition 1. *Let $F_1, F_2, \dots, F_\ell$ be vertex-disjoint rooted forests (i.e. each component has exactly one root), such that $e(F_i) > 0$ for some i. We call the tuple $F = (F_1, F_2, \dots, F_\ell)$ a rooted forest-tuple, and define $V(F) = \bigcup_{i\in[\ell]} V(F_i)$, $v(F) = |V(F)|$, $E(F) = \bigcup_{i\in[\ell]} E(F_i)$, $e(F) = |E(F)|$, and*

$$\begin{aligned} a(F) &= \max\{e(F_i):\ i \in [\ell]\},\\ k(F) &= |\{i \in [\ell]:\ e(F_i) = a(F)\}|,\\ t(F) &= |\{i \in [\ell] : e(F_i) > 0\}|. \end{aligned}$$

Moreover, we define the following:

(i) Let $\ell' \leqslant \ell$. We call a rooted forest-tuple $F' = (F'_1, F'_2, \dots, F'_{\ell'})$ a valid subforest of F if for every $i \in [\ell']$, we have $F'_i \subseteq F_i$ and every root of F'_i is a root of F_i. Furthermore, we assume that a rooted forest with no edges consisting of ℓ roots of F is also a valid subforest of F.

(ii) For a valid subforest $F' = (F'_1, F'_2, \dots, F'_\ell)$ of F, we define $F - F' := (F^*_1, F^*_2, \dots, F^*_\ell)$ to be a rooted forest-tuple in which for every $i \in [\ell]$, F^*_i is a rooted forest such that $V(F^*_i) = V(F_i)$, $E(F^*_i) = E(F_i)\backslash E(F'_i)$ and the root set of F^*_i is $V(F'_i)$. If e is an edge incident to a root in F_j, then by $F - e$ we mean the rooted forest-tuple $F - F'$, where F'_j has only one edge e and the remaining forests of F' have no edges.

(iii) We say that F is suitable if $3a(F) + k(F) - e(F) \leqslant 2$ or if $a(F) = 1$ and $t(F) \geqslant 4$.

(iv) We say that F is m-suitable if it is a valid subforest of a suitable forest-tuple F' with $e(F') = m$.

(v) We say that F is simple if $t(F) \geqslant 4$ and either $a(F) = 2$ and $k(F) = 1$, or $a(F) = 1$.

Our main goal is to analyze whether Waiter can force a red copy of a forest F, based only on the sizes of the trees in F. We will show that if F (or rather a forest-tuple made from F) is m-suitable, which roughly speaking means that every tree in F has less than $m/3$ edges and $e(F) \leqslant m$, then Waiter can force a red copy of F on the board K_{m+r}, where $r = v(F) - e(F)$ is the number of roots. However, in some cases it is useful to have some additional properties under control, for example the number of blue edges between groups of red trees. This is why we do not simply divide a forest F into trees, but rather into forests. Consequently, we will treat F as a forest-tuple and consider sizes of the tuple forests instead of sizes of the trees.

With a slight abuse of notation we will sometimes denote by F also the rooted graph $F_1 \cup F_2 \cup \ldots \cup F_\ell$ corresponding to the rooted forest-tuple $F = (F_1, F_2, \dots, F_\ell)$. Before we proceed, let us focus on some properties of suitable and m-suitable rooted forest-tuples.

Lemma 3.4. *If a rooted forest-tuple F is suitable, then $t(F) \geqslant 4$.*

Proof. Suppose that F is suitable and $t(F) \leqslant 3$. Then $e(F) \geqslant 3a(F) + k(F) - 2$. However, there are $k(F)$ forests with $a(F)$ edges and the remaining ones have at most $a(F) - 1$ edges. Hence

$$e(F) \leqslant k(F)a(F) + (3 - k(F))(a(F) - 1) = 3a(F) + k(F) - 3$$

and we reach a contradiction. □

Lemma 3.5. *If a rooted forest-tuple F satisfies $a(F) < e(F)/3$, then F is suitable.*

Proof. If $k(F) \leqslant 3$, then $3a(F) + k(F) - e(F) \leqslant 3a(F) + 3 - (3a(F) + 1) = 2$, so F is suitable. Hence, suppose that $k(F) \geqslant 4$. Using the fact that $e(F) \geqslant a(F)k(F)$ and $1 - a(F) \leqslant 0$, we get

$$3a(F) + k(F) - e(F) \leqslant 3a(F) + k(F) - a(F)k(F) = (k(F) - 3)(1 - a(F)) + 3 \leqslant 4 - a(F).$$

If $a(F) \geqslant 2$, then the inequality $3a(F) + k(F) - e(F) \leqslant 2$ holds. Otherwise we have $a(F) = 1$ and $t(F) = e(F) > 3$ by the assumption of the Lemma. Hence, in both cases F is suitable. □

Lemma 3.6. *Let $m \in \mathbb{N}$ and F be a rooted forest-tuple.*

- *If F is m-suitable, then $3a(F) + k(F) - m \leqslant 2$ or $a(F) = 1$.*
- *If $m \geqslant \max\{4, e(F)\}$ and either $3a(F) + k(F) - m \leqslant 2$ or $a(F) = 1$, then F is m-suitable.*

Proof. Suppose first that F is m-suitable. Then there exists a forest-tuple $F' \supseteq F$ with m edges satisfying $3a(F') + k(F') - m \leqslant 2$ or $a(F') = 1$. If $a(F') = 1$, then $a(F) \leqslant a(F') = 1$ and we are done.

Hence assume that $3a(F') + k(F') - m \leqslant 2$. If $a(F) = a(F')$, then $k(F) \leqslant k(F')$, so $3a(F) + k(F) - m \leqslant 3a(F') + k(F') - m \leqslant 2$. Therefore suppose that $a(F) < a(F')$. If $k(F) \geqslant 4$, then $3a(F) < k(F)a(F) \leqslant e(F)$, therefore, by Lemma 3.5, F is suitable and we have $3a(F) + k(F) - m \leqslant 2$ or $a(F) = 1$. Finally, if $k(F) < 4$, then $3a(F) + k(F) - m \leqslant 3a(F') + 1 - m \leqslant 3a(F') + k(F') - m \leqslant 2$ and we are done.

Now suppose that F satisfies $3a(F) + k(F) - m \leqslant 2$ or $a(F) = 1$, and let F' be a forest-tuple obtained from F by adding $m - e(F)$ forests, each consisting of a single edge. Then F is a valid subforest of F' so it is enough to show that F' is suitable. Assume first that $3a(F) + k(F) - m \leqslant 2$ and $a(F) > 1$. Then we have $a(F') = a(F)$, $k(F') = k(F)$ and $e(F') = m$, so $3a(F') + k(F') - e(F') = 3a(F) + k(F) - m \leqslant 2$, and hence F' is suitable. On the other hand, if $a(F) = 1$, then since $m \geqslant 4$, we have $t(F') \geqslant 4$, which implies that F' is suitable. □

Lemma 3.7. *Let F be an m-suitable rooted forest-tuple. Let F' be a rooted forest-tuple obtained from F by adding $m - e(F)$ additional forests, each consisting of a single edge. Then F' is suitable.*

Proof. Notice that $m \geqslant e(F)$ and $a(F) = a(F')$.

First consider the case $a(F) = 1$. Let F'' be a suitable rooted forest-tuple with $e(F'') = m$, given by m-suitability of F. It follows from Lemma 3.4 that $t(F'') \geqslant 4$. Since $a(F') = 1$ and $e(F') = m$, we have $t(F') \geqslant t(F'')$, and hence we get that F' is suitable as well.

Now suppose that $a(F) > 1$. We have $k(F) = k(F')$. Using Lemma 3.6 for F, we conclude that $3a(F') + k(F') - m = 3a(F) + k(F) - m \leqslant 2$. Thus F' is suitable. □

Lemma 3.8. *Let F be a rooted forest-tuple and $m \in \mathbb{N}$. If $a(F) < m/3$ and $m \geqslant e(F)$, then F is m-suitable.*

Proof. Let F' be a rooted forest-tuple obtained from F by adding $m - e(F)$ forests, each consisting of a single edge. In particular, if $e(F) = m$ then $F = F'$. We have $a(F') = a(F) < m/3 = e(F')/3$. Hence, by Lemma 3.5, F' is suitable. Since F is a valid subforest of F', the assertion follows. □

3.3. **Shrinking operations.** While considering a game in which Waiter is trying to obtain a red copy of a forest F (or a rooted forest-tuple) by increasing its partial embedding, one can look at the game as a process of decreasing the part of F left for embedding. Below we define two corresponding kinds of shrinking operations on rooted forest-tuples.

Definition 2. *Let $H = (H_1, H_2, \ldots, H_\ell)$ be a rooted forest-tuple with the set of roots R. Assume that $e(H_1) \geqslant e(H_2) \geqslant \ldots \geqslant e(H_\ell)$. Let B be a complete graph on at least $v(H)$ vertices, with fixed distinct images $u(r)$ of roots $r \in R$.*

(1) Suppose that $t(H) \geqslant 2$ and $r_i \in V(H_i)$, for $i = 1, 2$, are any non-isolated roots. We say that playing on B, Waiter performs an H-shrink operation of type 1 if he plays one round in the following way. Waiter offers the edges $u(r_1)w$ and $u(r_2)w$ for any free vertex $w \in V(B)$ which is not an image of a root of H.

(2) Suppose that $t(H) \geqslant 3$ and $r_i \in V(H_i)$, for $i = 1, 2, 3$, are any non-isolated roots. We say that playing on B, Waiter performs an H-shrink operation of type 2 if he plays two rounds in the following way. First Waiter offers the edges $u(r_1)w_1$ and $u(r_1)w_2$ for any distinct free vertices $w_1, w_2 \in V(B)$ which are not root images. Client colors $u(r_1)w_i$ blue and $u(r_1)w_j$ red. In the following round Waiter offers $u(r_2)w_i$ and $u(r_3)w_i$.

During the above H-shrink operation of type 1, Client claims a red edge e incident to the image $u(r)$ of a root $r \in V(H_i)$, for some $i \in \{1, 2\}$. Let e' be any edge of H incident to r. We say that the rooted forest-tuple $H - e'$ arises from the H-shrink operation of type 1*. Analogously, we define a rooted forest-tuple arising from the H-shrink operation of type 2, namely, it is $(H - e_1') - e_2'$, where e_1', e_2' are edges of H corresponding to the two red edges claimed by Client.*

For simplicity of the above definition, we assumed the forests of H are sorted by their number of edges. However, we will also allow H-shrink operations in case of any rooted forest-tuple – in an H-shrink operation of type 1 Waiter deals with any two biggest forests of the tuple, while in an H-shrink operation of type 2 she deals with any three biggest forests.

In the following lemma we prove among others that H-shrink operations of both types preserve suitability, provided H is not simple.

Lemma 3.9. *Let $H = (H_1, H_2, \ldots, H_\ell)$ be a rooted forest-tuple with the set of roots R and $s \in \{1, 2\}$. Suppose that* $\min(k(H), 2) = 3 - s$ *and $t(H) > s$. Let B be a complete graph on at least $v(H)$ vertices,* *with fixed distinct images $u(r)$ of roots $r \in R$. Assume that playing on B, Waiter performs an H-shrink operation of type s and let H' be the rooted forest-tuple arising from this operation. Then the following holds.*

(a) If H is suitable and not simple, then H' is suitable.

(b) If H is m-suitable and $e(H) - t(H) \geqslant 2$, then H' is $(m-s)$-suitable.

Proof. Since the order of forests in F is irrelevant for the above properties, we can assume that $e(H_1) \geqslant e(H_2) \geqslant e(H_3) \geqslant \ldots \geqslant e(H_\ell)$. Suppose that H is m-suitable and $e(H) - t(H) \geqslant 2$. Then $a(H) \geqslant 2$ and hence $3a(H) + k(H) - m \leqslant 2$ in view of Lemma 3.6. Another consequence of the inequality $e(H) - t(H) \geqslant 2$ is that $3a(H) + k(H) \geqslant 8$, so $m \geqslant 6$. Thus, we also have $m - s \geqslant 4$. We know that $e(H') = e(H) - s$ and $m \geqslant e(H)$, hence $m - s \geqslant \max\{4, e(H')\}$. Therefore, in order to prove that H' is $(m-s)$-suitable, it is enough to verify that $3a(H') + k(H') - (m-s) \leqslant 2$ and use Lemma 3.6. We consider two cases depending on s.

Case 1. Suppose that $s = 1$. Then $k(H) \geqslant 2$ by assumption that $\min(k(H), 2) = 3 - s = 2$, and hence $e(H_1) = e(H_2) = a(H)$. Observe that the H-shrink operation of type 1 decreases either $e(H_1)$ or $e(H_2)$ by 1, so $a(H') = a(H)$ and $k(H') = k(H) - 1$. Thus

$$3a(H') + k(H') - (m-s) = 3a(H) + k(H) - m \leqslant 2.$$

Case 2. Suppose that $s = 2$. Then $k(H) = 1$ by assumption that $\min(k(H), 2) = 3 - s = 1$. The H-shrink operation of type 2 decreases $e(H_1)$ and exactly one of $e(H_2), e(H_3)$ by 1. We note that $a(H') = a(H) - 1$ and consider two subcases, depending on the structure of H'.

- If $k(H') \leqslant 2$, then

$$3a(H') + k(H') - (m-s) \leqslant 3(a(H)-1) + 2 - (m-2) = 3a(H) + k(H) - m \leqslant 2.$$

- If $k(H') \geqslant 3$, then $e(H_2) = e(H_3) = \ldots = e(H_{k(H')+1}) = a(H) - 1$, so $a(H) + (a(H)-1)k(H') \leqslant e(H) \leqslant m$. This implies that

$$\begin{aligned} 3a(H') + k(H') - (m-s) - 2 &= 3(a(H)-1) + k(H') - m \\ &\leqslant 3a(H) - 3 + k(H') - a(H) - (a(H)-1)k(H') \\ &= 1 - (a(H)-2)(k(H')-2). \end{aligned}$$

  Since $e(H) - t(H) \geqslant 2$ and $k(H) = 1$, we have $a(H) \geqslant 3$, so $1 - (a(H)-2)(k(H')-2) \leqslant 0$.

In both subcases $3a(H') + k(H') - (m-s) \leqslant 2$ and the assertion (b) follows.

Property (a) is a consequence of (b) applied with $m = e(H)$, since if H is suitable and not simple, then $e(H) - t(H) \geqslant 2$. □

Remark 3.10. *If F is suitable, then $t(F) \geqslant 4 > 2 \geqslant s$. Therefore, in this case we do not need the assumption $t(F) > s$, required for performing a shrink operation of type s.*

Shrinking operations will be the main ingredient of Waiter's strategy in most games considered further in this chapter, but it would be more convenient to look at them in the reversed way, i.e. increasing the red graph. Below we define an operation of type s corresponding to a shrink operation of type s.

Let $F = (F_1, F_2, \ldots, F_\ell)$ be a rooted forest-tuple with the set of roots R. Consider a game played on a board B which is the complete graph, and such that Waiter's aim is to force a red copy of F. Suppose that after a number of rounds Client's graph is a red copy $\bar{F}' = \bar{F}'_1 \mathbin{\dot\cup} \bar{F}'_2 \mathbin{\dot\cup} \ldots \mathbin{\dot\cup} \bar{F}'_\ell$ of a valid subforest $F' = (F'_1, F'_2, \ldots, F'_\ell)$ of F. Let R' be the set of roots of $F - F'$, and for every $r \in R'$ let $u(r)$ be the image of r in $\bar{F}'$. We say that Waiter performs an *operation of type* $s \in \{1, 2\}$, if she performs an $(F - F')$-shrink operation of type s, according to Definition 2 applied with fixed vertices $u(r)$, $r \in R'$. During the operation of type 2, Client claims red edges e_1, e_2 incident to roots $r_1 \in V(\bar{F}'_i)$, $r_2 \in V(\bar{F}'_j)$, $i \neq j$, respectively. We define $\bar{F}''_i = \bar{F}'_i \cup \{e_1\}$, $\bar{F}''_j = \bar{F}'_j \cup \{e_2\}$ and $\bar{F}''_k = \bar{F}'_k$ for $k \notin \{i, j\}$, and we call $\bar{F}'' = \bar{F}''_1 \mathbin{\dot\cup} \bar{F}''_2 \mathbin{\dot\cup} \ldots \mathbin{\dot\cup} \bar{F}''_\ell$ the red graph arising from the operation of type 2. We define the red graph arising from the operation of type 1, in which $\bar{F}'$ increases by one edge, in an analogous way. Note that if $\bar{F}''$ arises from the operation of type s, then it is a copy of a valid subforest F'' of F such that $F - F''$ arises from the $(F - F')$-shrink operation of type s.

Observe that Waiter is able to perform operation of type $s \in \{1, 2\}$ if Client's graph $\bar{F}'$ satisfies $t(F - F') > s$ and there are at least s free vertices on the board which are not the roots of $\bar{F}'$. Furthermore, if Waiter is going to increase a red forest $\bar{F}_i$ by the operation of type 1, then she has freedom in deciding which edge of F_i, not yet embedded, will correspond to the next red edge e in a copy of F_i, provided an endpoint of e is mapped into a vertex of $\bar{F}'_i$. Similarly, while increasing $\bar{F}_i$ or $\bar{F}_j$ by the operation of type 2, Waiter is flexible in selecting (simultaneously) edges $e_i \in E(F_i), e_j \in E(F_j)$ such that one of them will correspond to the next red edge in a copy of F_i or F_j, as long as e_i, e_j have exactly one endpoint mapped into a vertex of $\bar{F}_i, \bar{F}_j$, respectively. We summarize this observation in the following remark used later in the proofs.

Remark 3.11. *Consider a game in which Waiter's aim is to obtain a red copy of a rooted forest-tuple $F = (F_1, F_2, \ldots, F_\ell)$. Suppose she can force edges of F_i in a series of (not necessarily subsequent) operations of type 1 or 2. Then the edges of F_i can be forced by Waiter in any order, provided that at every stage of the game the part of F already embedded is a valid subforest of F. Such an ordering of embedding edges can be fixed for every F_i at the beginning of the game.*

The result below is a straightforward consequence of the definitions of a valid subforest, the operation of type s and Lemma 3.9, so we omit its proof.

Lemma 3.12. *Let $F = (F_1, F_2, \ldots, F_\ell)$ be a rooted forest-tuple with its root set R and let $F' = (F'_1, F'_2, \ldots, F'_\ell)$ be a valid subforest of F. Consider a moment of the game played on the complete graph B such that Client's graph is a red copy $\bar{F}' = \bar{F}'_1 \mathbin{\dot\cup} \bar{F}'_2 \mathbin{\dot\cup} \ldots \mathbin{\dot\cup} \bar{F}'_\ell$ of F'. If Waiter is able to perform the operation of type $s \in \{1, 2\}$, then the following holds for the red graph $\bar{F}''$ arising from this operation.*

(a) $\bar{F}''$ is a copy of a valid subforest $F'' = (F''_1, F''_2, \ldots, F''_\ell)$.

(b) No edge claimed during the operation of type s has both endpoints in the roots of F.

(c) Every blue edge claimed during the operation of type s lies between sets $V(\bar{F}''_i)$ and $V(\bar{F}''_j)$ for some $i \neq j$.

(d) Every red edge claimed during the operation of type s is an edge of $\bar{F}''_i$, for some $i \in [\ell]$.

(e) If $F - F'$ is suitable but not simple and $\min(k(F - F'), 2) = 3 - s$, then $F - F''$ is suitable.
(f) If $F - F'$ is m-suitable, $e(F) - t(F) \geqslant 2$, and $\min(k(F - F'), 2) = 3 - s$, then $F - F''$ is $(m - s)$-suitable.

3.4. **Forcing rooted forest-tuples.** We are ready to prove a couple of results regarding forcing rooted forests.

Lemma 3.13. *Let $F = (F_1, F_2, \dots, F_\ell)$ be a suitable rooted forest-tuple with the set of roots R. Let B be a complete graph on $v(F)$ vertices, with fixed distinct vertices $u_1, u_2, \dots, u_{|R|}$. Then, playing on B, Waiter can force a red copy $\bar{F}' = \bar{F}_1' \cup \bar{F}_2' \cup \dots \cup \bar{F}_\ell'$ of a valid subforest $F' = (F_1', F_2', \dots, F_\ell')$ of F such that $F - F'$ is suitable and simple. Furthermore, she can achieve this goal within $e(F')$ rounds applying operations of type 1 or 2 and such that after the j^{th} operation Client's graph is a red copy $\bar{F}^j = \bar{F}_1^j \cup \bar{F}_2^j \cup \dots \cup \bar{F}_\ell^j$ of a valid subforest $F^j = (F_1^j, F_2^j, \dots, F_\ell^j)$ of F satisfying the following properties.*

(a) For every $r \in R$, the image of r in $\bar{F}^j$ is u_r.
(b) No edges between the images of the roots are colored.
(c) Every blue edge lies between the sets $V(\bar{F}_i^j)$ and $V(\bar{F}_{i'}^j)$, for some $i \neq i'$.
(d) Every red edge is an edge of $\bar{F}^j$.
(e) $F - F^j$ is suitable.

Proof. Note that there are exactly $|R|$ trees which are the components of F. The general strategy for Waiter is to force Client to grow each tree starting from the root, until the obtained rooted forest-tuple is simple. Our argument is inductive.

At the beginning of the game, let F_i^0 consist of the roots in $R \cap V(F_i)$, and $\bar{F}_i^0$ consist of their fixed images u_r, $r \in R \cap V(F_i)$. Then $\bar{F}^0 := (\bar{F}_1^0, \bar{F}_2^0, \dots, \bar{F}_\ell^0)$ and $F^0 := (F_1^0, F_2^0, \dots, F_\ell^0)$ satisfy (a) – (e).

For $j \geqslant 0$, assume that j operations of type 1 or 2 were already played and that Client's graph $\bar{F}_1^j \cup \bar{F}_2^j \cup \dots \cup \bar{F}_\ell^j$ is a copy of a rooted forest $F_1^j \cup F_2^j \cup \dots \cup F_\ell^j$ as described above. Assume further that $F - F^j$ is not simple and hence Waiter is going to play the $(j + 1)^{\text{st}}$ operation of type 1 or 2. Note that $F - F^j$ is suitable, so $t(F - F^j) \geqslant 4$ and there are at least 4 free vertices among non-roots images on the board, so Waiter can perform an operation of any type. The type of the operation depends on $k(F - F^j)$. If $k(F - F^j) \geqslant 2$, then she performs the operation of type 1; otherwise she performs the operation of type 2. Let $\bar{F}^{j+1}$ be the red graph arising from this operation. In view of Lemma 3.12 applied to F and F^j, the graph $\bar{F}^{j+1}$ is a copy of a valid subforest of F, let us denote it by F^{j+1}. Conditions (a) – (e) are satisfied by $\bar{F}^{j+1}$ and F^{j+1} in view of the inductive hypothesis and Lemma 3.12. In particular, the suitability of F^{j+1} follows from Lemma 3.12 (e).

Observe that the number of edges of $F - F^j$ is decreasing with j, while $t(F - F^j) \geqslant 4$ by Lemma 3.4, so $F - F^s$ is simple for some $s \geqslant 0$. It is also suitable because of (e). Finally, note that every red edge claimed in the game becomes an edge of $\bar{F}^s$, so it takes Waiter $e(F^s)$ rounds in total to achieve the desired graph. □

Lemma 3.14. *Let $F = (F_1, F_2, \ldots, F_\ell)$ be a suitable rooted forest-tuple. Then playing on a complete graph B with $v(F)$ vertices, Waiter can force a copy $\bar{F} = \bar{F}_1 \mathbin{\dot\cup} \bar{F}_2 \mathbin{\dot\cup} \ldots \mathbin{\dot\cup} \bar{F}_\ell$ of $F_1 \mathbin{\dot\cup} F_2 \mathbin{\dot\cup} \ldots \mathbin{\dot\cup} F_\ell$ within $e(F) + 1$ rounds and such that the following properties hold.*

(a) All roots are mapped into vertices fixed at the beginning.
(b) No edges between the images of the roots are colored.
(c) For every $i \in [\ell]$ there are no blue edges spanned by vertices in $V(\bar{F}_i)$.
(d) There are no red edges spanned by vertices in $V(\bar{F}_i)$ other than those of $\bar{F}_i$.

Proof. Let R be the set of all roots of $F_1 \mathbin{\dot\cup} F_2 \mathbin{\dot\cup} \ldots \mathbin{\dot\cup} F_\ell$, and for each $r \in R$ fix a distinct vertex u_r of B. We present Waiter's strategy in two stages. In the first stage she forces Client to build a copy of a valid subforest F' of F such that $F - F'$ is simple. In the second stage Waiter forces a copy of $F - F'$, which completes building a copy of F.

Stage I. Waiter applies her strategy provided by Lemma 3.13 and forces Client to build a red copy $\bar{F}' = (\bar{F}'_1, \bar{F}'_2, \ldots, \bar{F}'_\ell)$ of a valid subforest $F' = (F'_1, F'_2, \ldots, F'_\ell)$ of F such that $F - F'$ is suitable and simple, and such that every root u_r of $\bar{F}'$ is the image of r, for $r \in R$. It follows from Lemma 3.13 that no edges between the roots u_r are colored, all red edges are edges of $\bar{F}'$ and every blue edge lies between the sets $V(\bar{F}'_i)$ and $V(\bar{F}'_j)$, for some $i \neq j$. The first stage lasts $e(F')$ rounds.

Stage II. Let $F^* = (F^*_1, \ldots, F^*_\ell) := F - F'$. We set $a_i = e(F^*_i) = e(F_i) - e(F'_i)$ for every $i \in [\ell]$. By reordering forests $F'_1, \ldots, F'_\ell$, we can assume that $a_1 \leqslant 2$ and $a_2 = \ldots = a_{\tilde{\ell}} = 1$ with $\tilde{\ell} := t(F - F') \geqslant 4$, since $F - F'$ is simple and suitable.

First assume that $a_1 = 2$. Let $V(B) \backslash V(\bar{F}') =: \{v_1, v_2, \ldots, v_{\tilde{\ell}+1}\}$. Notice that all vertices of this set are free because of the properties of $\bar{F}'$ mentioned at the end of Stage I.

Waiter now plays as follows. For the first round of Stage II, let w be the image of a root of F^*_1. Waiter offers $wv_{\tilde{\ell}}$ and $wv_{\tilde{\ell}+1}$. By symmetry, we can assume that Client colors $wv_{\tilde{\ell}+1}$ red.

Afterwards, for every $i \in [\tilde{\ell}]$, let $w_i \in V(\bar{F}'_i)$ be the unique vertex at which still exactly one red edge needs to be forced. Then Waiter's goal is to get a red perfect matching between the sets $W := \{w_1, w_2, \ldots, w_{\tilde{\ell}}\}$ and $V := \{v_1, v_2, \ldots, v_{\tilde{\ell}}\}$. Due to the previous round, in the following rounds Waiter cannot offer $w_1 v_{\tilde{\ell}}$, since either $w_1 = w$ and $w_1 v_{\tilde{\ell}}$ is blue or $w_1 \neq w$ and $w_1 v_{\tilde{\ell}}$ is free but, if colored red, would violate condition (c). Other edges between W and V are still free. Hence, using Lemma 3.3, Waiter can force a required red matching within $\tilde{\ell} + 1$ rounds.

Now assume that $a_1 = 1$. This case is similar, yet a bit simpler, since we can apply Lemma 3.3 at once.

In both cases Waiter wastes one red edge when she tries to force a perfect matching, so Stage II lasts $e(F) - e(F') + 1$ rounds. Furthermore, this extra red edge joins copies of two distinct forests F_i and F_j, and every blue edge claimed in Stage II joins copies of two distinct forests as well.

Summarizing, Stage I and Stage II last together $e(F) + 1$ rounds and all properties (a) – (d) are satisfied. □

In some cases Waiter may want to force a forest that has a smaller number of vertices than the board B. We formulate two lemmas, corresponding to Lemma 3.13 and Lemma 3.14, with an extra condition on the blue edges incident to vertices not belonging to the red forest obtained at the end.

Lemma 3.15. *Let $F = (F_1, F_2, \ldots, F_\ell)$ be an m-suitable rooted forest-tuple with the set of roots R. Let B be a complete graph on $v(F) + m - e(F)$ vertices, with fixed ℓ vertices $u_1, u_2, \ldots, u_\ell$. Then, playing on B, Waiter can force a red copy $\bar{F}' = \bar{F}'_1 \cup \bar{F}'_2 \cup \ldots \cup \bar{F}'_\ell$ of a valid subforest $F' = (F'_1, F'_2, \ldots, F'_\ell)$ of F such that $F - F'$ is $(m - e(F'))$-suitable and either $t(F - F') \leqslant 2$ and $k(F - F') = 1$, or $e(F - F') - t(F - F') \leqslant 1$. Furthermore, she can achieve this goal within $e(F')$ rounds applying operations of type 1 or 2, and such that after the j^{th} operation, Client's graph is a red copy $\bar{F}^j = \bar{F}^j_1 \cup \bar{F}^j_2 \cup \ldots \cup \bar{F}^j_\ell$ of a valid subforest $F^j = (F^j_1, F^j_2, \ldots, F^j_\ell)$ of F satisfying the following properties.*

(a) For every $r \in R$, the image of r in $\bar{F}^j$ is u_r.

(b) No edges between the images of the roots are colored.

(c) Every blue edge lies between the sets $V(\bar{F}^j_i)$ and $V(\bar{F}^j_{i'})$, for some $i \neq i'$.

(d) Every red edge is an edge of $\bar{F}^j$.

(e) $F - F^j$ is $(m - e(F^j))$-suitable.

Proof. The argument is similar to the proof of Lemma 3.13, nonetheless we present the inductive argument (omitting a few details), since an m-suitable rooted forest-tuple may not be suitable.

For $j \geqslant 0$, assume that j operations of type 1 or 2 were already played and that Client's graph $\bar{F}^j_1 \cup \bar{F}^j_2 \cup \ldots \cup \bar{F}^j_\ell$ is a copy of a rooted forest $F^j_1 \cup F^j_2 \cup \ldots \cup F^j_\ell$ satisfying (a) – (e). Assume further that $e(F - F^j) - t(F - F^j) \geqslant 2$, and $t(F - F^j) \geqslant 3$ or $k(F - F^j) \geqslant 2$, so Waiter is going to perform the $(j+1)^{\text{st}}$ operation of type 1 or 2. If $k(F - F^j) \geqslant 2$, then Waiter performs the operation of type 1, while if $k(F - F^j) = 1$ and $t(F - F^j) \geqslant 3$, then the operation of type 2 is performed. Let $\bar{F}^{j+1}$ be the red graph arising from the operation of type s, $s \in \{1, 2\}$, performed by Waiter. In view of Lemma 3.12 applied to F and F^j, the graph $\bar{F}^{j+1}$ is a copy of a valid subforest of F. Let us denote it by F^{j+1}. Conditions (a) – (e) are satisfied by $\bar{F}^{j+1}$ and F^{j+1} in view of the inductive hypothesis and Lemma 3.12. In particular, the $(m - e(F^{j+1}))$-suitability of F^{j+1} follows from Lemma 3.12 (f), the assumption that $F - F^j$ is $(m - e(F^j))$-suitable, $e(F - F^j) - t(F - F^j) \geqslant 2$, and $m - e(F^j) - s = m - e(F^{j+1})$. This finishes the inductive argument. □

Let us recall that if a vertex u of a rooted forest F is a root, then we do not treat it as a leaf, even if $\deg_F(u) = 1$. For example, an isolated edge in a rooted forest has only one leaf, not two.

Lemma 3.16. *Let $F = (F_1, F_2, \ldots, F_\ell)$ be an m-suitable rooted forest-tuple and let $m > e(F)$. Then playing on a complete graph B on $m + v(F) - e(F)$ vertices, Waiter can force a red copy $\bar{F} = \bar{F}_1 \cup \bar{F}_2 \cup \ldots \cup \bar{F}_\ell$ of $F_1 \cup F_2 \cup \ldots \cup F_\ell$ within $e(F)$ rounds and such that the following properties hold.*

(a) All roots are mapped into vertices fixed at the beginning.

(b) No edges between the images of the roots are colored.

(c) For every $i \in [\ell]$, there are no blue edges spanned by vertices in $V(\bar{F}_i)$.

(d) There are no red edges spanned by vertices in $V(\bar{F}_i)$ other than those of $\bar{F}_i$.

(e) All colored edges intersect $V(\bar{F})$.

(f) Every vertex not in $V(\bar{F})$ is incident with at most one colored edge, and the other endpoint of such an edge is not a leaf of $\bar{F}$.

Proof. We divide the game into two stages. In the first stage Waiter forces a red copy of a valid subforest F' of F such that $F - F'$ either has at most two non-empty forests or all forest in the tuple are very small. In the second stage, Waiter forces a copy of $F - F'$, which completes building a copy of F.

Stage I. Waiter applies her strategy provided by Lemma 3.15 and forces Client to build a red copy $\bar{F}' = (\bar{F}_1', \bar{F}_2', \ldots, \bar{F}_\ell')$ of a valid subforest $F' = (F_1', F_2', \ldots, F_\ell')$ of F such that $F^* := F - F'$ is $(m - e(F'))$-suitable and either $t(F^*) \leqslant 2$ and $k(F^*) = 1$, or $e(F^*) - t(F^*) \leqslant 1$. Furthermore, every root u_r of $\bar{F}'$ is the image of r, for $r \in R$. It follows from Lemma 3.15 that no edges between the vertices u_r are colored, all red edges are edges of $\bar{F}'$ and every blue edge lies between sets $V(\bar{F}_i')$ and $V(\bar{F}_j')$ for some $i \neq j$. The first stage lasts $e(F')$ rounds.

Stage II. For $F^* = (F_1^*, \ldots, F_\ell^*)$ we set $a_i = e(F_i^*) = e(F_i) - e(F_i')$ for every $i \in [\ell]$. By reordering the forests $F_1^*, \ldots, F_\ell^*$, we can assume that $a_1 \geqslant a_2 \geqslant \ldots \geqslant a_\ell$. We consider two cases depending on the properties of F^* mentioned at the end of Stage I.

Case 1. Suppose that $t(F^*) \leqslant 2$ and $k(F^*) = 1$. Then $a_1 = a(F^*)$ and $a_2 \leqslant a(F^*) - 1$ (in case $\ell = 1$ we set $a_2 = 0$).

If $a_1 = 1$, then Waiter can simply finish the game in one round, since $v(B) = v(F) + m - e(F) \geqslant v(F) + 1 = v(F^*) + 2$, and hence there are still two free vertices (which are not roots) at this moment. Here we used the assumption that $m > e(F)$.

If $a_1 > 1$, then in view of $(m - e(F'))$-suitability of F^* and Lemma 3.6, we have $3a(F^*) + k(F^*) - (m - e(F')) \leqslant 2$. Thus $m \geqslant 3a(F^*) - 1 + e(F')$ and the number of free vertices (which are not roots) is $v(B) - v(F') = m - e(F) + v(F) - v(F') \geqslant 3a(F^*) - 1 \geqslant 2a_1 + a_2$.

For the next a_2 rounds, Waiter performs the operation of type 1. Let $\bar{F}''$ be the red graph arising after all a_2 operations. It follows from Lemma 3.12 and the properties of the colored graph at the end of Stage I that no edges between the vertices u_r are colored, all red edges are edges of $\bar{F}''$ and every blue edge lies between distinct forests of $\bar{F}''$. Note that exactly a_2 free vertices where used in these a_2 rounds.

The part of F remaining for embedding has a_1 edges and since there are still at least $2a_1$ free vertices left, Waiter can simply use two free vertices for each remaining edge to force the desired red copy $\bar{F}$. Note that the only vertices not in $V(\bar{F})$ incident to a colored edge at the end of the game are the ones used during the last a_1 rounds. Every such vertex was used exactly once and the edge incident to that vertex was also incident to a vertex which was already in the red forest and which had another red incident edge, so it was not a leaf. Therefore properties (e) and (f) are satisfied. Since the last a_1 rounds do not spoil properties (a) – (d), we conclude that properties (a) – (f) hold.

Case 2. Suppose that Case 1 does not hold and $e(F^*) - t(F^*) \leqslant 1$.

Then either $a_1 = 2$, $t(F^*) \geqslant 3$ and $a_2, a_3, \ldots, a_\ell \leqslant 1$, or $a_1 = 1$. In the first case Waiter can perform the operation of type 2. The red graph $\bar{F}''$ arising after this operation satisfies $a(\bar{F}'') = 1$ and further argument coincides with the analysis in case $a_1 = 1$. For the next $t(F^*) - 1$ rounds, Waiter performs the operation of type 1. Thereby she forces a red matching of size $t(F^*) - 1$ and only one edge of F remains for embedding. Then Waiter can finish the game in one round, since there are two free vertices (which are not roots) at this moment. Since the operations of type 1 or 2 did not spoil properties (a) – (d), and the last round did not spoil properties (e) and (f), we obtain a desired red copy of F. □

We are now ready to state and prove one of the main results of this section.

Theorem 3.17. *Let $0 \leqslant q < t$ be integers, and let F be a forest with at most t edges, consisting of s rooted trees $T_1, T_2, \ldots, T_s$, and such that for each $i \in [s]$ we have $1 \leqslant e(T_i) < \frac{t-q}{3}$. Let $r_1, \ldots, r_s$ be the roots of the corresponding trees. Consider a partially colored complete graph B with $V(B) = U \cup V$, $|U| = s$, $|V| = t$, and such that B contains a collection of s vertex-disjoint blue stars with centers $u_1, \ldots, u_s \in U$, with total number of edges equal $q \leqslant t$, while all other edges of $B[U, V]$ and all edges of $B[V]$ are free. Then, playing on B, Waiter can force a red copy $\bar{F}$ of F within at most $e(F) + 1$ rounds in such a way that the following holds.*

- *(a) Every root r_i is mapped into u_i.*
- *(b) All colored edges of $\bar{F}$ intersect $V(\bar{F})$.*
- *(c) No edges connecting vertices from U are offered by Waiter.*
- *(d) Every vertex not in $V(\bar{F})$ is incident with at most one colored edge in $\bar{F}$, and the other endpoint of such an edge is not a leaf of $\bar{F}$.*

Proof. We apply induction on q.

For $q = 0$ and $e(F) = t$ we have $a(F) < e(F)/3$. Hence, by Lemma 3.5, F is suitable, and by Lemma 3.14, Waiter can force the desired copy of $\bar{F}$ in B.

For $q = 0$ and $e(F) < t$ we also have $a(F) < t/3$. Hence, using Lemma 3.8, we get that F is t-suitable. Then, by Lemma 3.16, Waiter can force the desired copy of $\bar{F}$ in B.

For $q > 0$ we consider two cases, in both of which we start by mapping the roots r_i to vertices u_i in U.

Case 1. $|U| \geqslant 3$.

Then there exists a vertex $v \in V$ which is incident to a blue edge and there are two root images $u_i, u_j \in U$, such that the edges u_iv, u_jv are free. Waiter offers u_iv, u_jv. By symmetry we can assume that Client colors u_iv red. Let T_i be the component of F with root r_i, and let r_i' be any neighbor of r_i in T_i. We replace F with $F' := F - r_ir_i'$, U with $U' := U \cup \{v\}$ and V with $V' := V \backslash \{v\}$, and add r_i' to the set of roots of F with a fixed image v. Moreover, if the new tree rooted at r_i or r_i' in F' is trivial, we delete it from F', and we delete the corresponding image of that root from U' as well. We are left with the forest F' with $t' = t - 1$ edges and a board $B' := B[U' \cup V']$ such that there are $q' = q - 1$ blue edges forming stars with centers in U' and edges towards V'. As every component of

F' has less than $\frac{t-q}{3} = \frac{t'-q'}{3}$ edges, we can apply the induction hypothesis. Indeed, note that the edges offered in the first round are not incident with V', and hence all edges in $B'[U', V']$ and $B'[V']$, apart from the $q-1$ initial blue edges, are free. This ensures that Waiter can force a copy of F' where each of the roots $r_1, \ldots, r_s, r'_i$ is mapped to its fixed image $u_1, \ldots, u_s, v$, respectively. Together with the red edge $u_i v$, we get the desired copy $\bar{F}$ of F.

Case 2. $|U| \leqslant 2$.

Then there are at most two trees in F, and hence $e(F) < \frac{2}{3}(t-q)$. In this case we can remove all vertices in V which are incident to a blue edge, thus obtaining a subset $V' \subset V$ of $t' := t - q$ vertices. Let $B' := B[U \cup V']$ be the new board. The smaller board B' has $s + t'$ vertices and no blue edges. Since $e(F) < \frac{2}{3}(t-q) \leqslant t'$ and each component T_i in F satisfies $e(T_i) < \frac{t-q}{3} = \frac{t'}{3}$, the assertion follows from the induction base case applied to the board B'. □

The last result of this subsection is an observation that if the board is much larger than the target forest, then Waiter can easily achieve her goal.

Lemma 3.18. *Let F be a rooted forest and let B be a complete graph on at least $v(F) + e(F)$ vertices. Then Waiter can force a red copy $\bar{F}$ of F in B within at most $e(F)$ rounds in such a way that*

(a) all roots are mapped into vertices fixed at the beginning,

(b) every colored edge has an endpoint in a non-leaf of $\bar{F}$,

(c) all blue edges in $E_B(V(\bar{F}), V(B)\backslash V(\bar{F}))$ form a star forest such that every star is centered in a vertex $u \in V(\bar{F})$ and has at most $deg_{\bar{F}}(u)$ edges.

Proof. Waiter has enough space to force a red copy of every component of F in a greedy way: starting from its roots, Waiter always offers two edges incident to a vertex of a partial embedding of F, with both other endpoints free. Then the properties (a), (b) and (c) are satisfied. □

Forcing forests with double-rooted components. We have dealt with rooted forests so far. In the remaining part of the section we analyze games in which Waiter's aim is to obtain a red forest with double-rooted components (trees). We abandon the forest-tuples notation while stating the theorems, but it will still be present in the proofs.

Lemma 3.19. *Let F be a forest such that each of its components is rooted or double-rooted. Assume that every tree in F has at least 1 and less than $\frac{e(F)-d}{3}$ edges, where d is the number of double-rooted components of F, and that in every double-rooted component its roots are at distance at least 4. Then playing on a complete graph B on $n \geqslant v(F)$ vertices, Waiter can force a red copy $\bar{F}$ of F within at most $e(F) + d + 1$ rounds in such a way that all roots are mapped into vertices fixed at the beginning.*

Proof. The main idea is to split every double-rooted tree into a forest consisting of two trees by removing one edge, and then follow the strategy from Lemma 3.14. The main strategy will be interrupted at some point in order to connect such pairs of trees into one.

We call a double-rooted tree T a *path-double-rooted* tree, if every inner vertex of a path from one root to the other has degree exactly 2 in T; in other words, the path connecting the roots in T is a bare path. Suppose that $F = T_1 \cup T_2 \cup \ldots \cup T_\ell$, where the first d_1 trees are double-rooted but not path-double-rooted, the next d_2 trees are path-double-rooted, and the remaining trees are rooted which means that they have exactly one root. Let R_1, R_2 and R_3 be the sets of roots of the above trees of three kinds, respectively, and let $V(B) = U_1 \cup U_2 \cup U_3 \cup V$, where U_i consists of the images of the roots in R_i, for $i \in [3]$.

Consider a double-rooted tree T_i which is not path-double-rooted, i.e. with $i \in [d_1]$. Suppose that r_i and r_i' are its two roots, P_i is a path connecting r_i to r_i', and let v be an inner vertex of degree at least 3 in T_i. Moreover, without loss of generality suppose that v and r_i' are at distance at least 2. Let x be the neighbor of v on a path to r_i' and let y be the neighbor of x on a path to r_i' (note that if v and r_i' are at distance 2, then $y = r_i'$). Finally, let w be a neighbor of v which does not belong to P_i. Set $T_i' := T[V(P_i) \cup \{w\}]$, $T_i^{(1)} \cup T_i^{(2)} := T_i' - \{xy\}$ (see Figure 3.2) and $F_i' := T_i - xy$. Furthermore, for every path-double-rooted tree T_j, i.e. with $j = d_1 + k$ for $k \in [d_2]$, let P_j be the path between the roots r_j and r_j' in T_j and let q be the neighbor of r_j' on P_j. We set $T_j' = P_j - \{r_j'\}$ and $F_j' := F_j - r_j'q$. Hence T_j' is just a rooted path where the root is a vertex of degree 1.

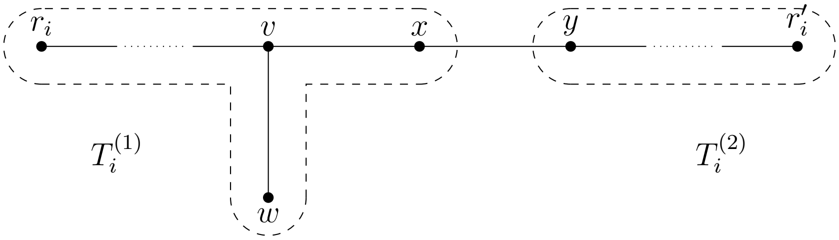

Figure 3.2. The subtree T_i' of the double-rooted tree T_i, which is not path-double-rooted. The forest F_i' contains trees $T_i^{(1)}$ and $T_i^{(2)}$, which are forced before the rest of T_i.

We now consider the rooted forest-tuple $F' = (F_1', F_2', \ldots, F_\ell')$, where for $h > d_1 + d_2$, we set $F_h' = T_h$. We claim that F' satisfies the assumptions of Lemma 3.14. We need to check whether F' is suitable. Since $e(T_i) \leqslant \frac{e(F)-d}{3}$ for any $i \in [\ell]$, and since we have removed from F exactly d edges, we have $a(F') \leqslant \frac{e(F)-d}{3} = \frac{e(F')}{3}$. Thus, by Lemma 3.5, F' is suitable.

Waiter now plays according to the strategy in the proof of Lemma 3.14, which will be interrupted in order to connect the roots in each image of a double-rooted tree as follows. Before we proceed, let us recall that the operations of type 1 or 2 are the heart of the proof of Lemma 3.14, so in view of Remark 3.11, Waiter has some flexibility while choosing the order of the edge embedding for a given forest. Hence, while forcing F_i' for $i \in [d_1]$, she can first force copies of $T_i^{(1)}$ and $T_i^{(2)}$, and while forcing F_j' for $j = d_1 + k$ with $k \in [d_2]$, she first forces T_j'.

Suppose that for a tree T_i which is double-rooted but not path-double-rooted, Waiter already forced in B its subforest $\bar{F}_i'$ (see Figure 3.2) in such a way, that the only colored edges spanned by vertices in $V(\bar{F}_i')$ are the red edges in $\bar{F}_i'$. In particular, the edges $\bar{y}\bar{x}$ and $\bar{y}\bar{w}$, where $\bar{x}, \bar{y}, \bar{w}$ are the images of x, y, w, respectively, are still free. Hence, Waiter offers $\bar{y}\bar{x}$ and $\bar{y}\bar{w}$. Now no matter which of the two edges Client chooses, Waiter forces a red copy $\bar{T}_i'$ of the double-rooted subtree $T_i' \subset T_i$, possibly changing the role of x and w. Waiter repeats this strategy for each tree T_i with $i \in [d_1]$ using exactly d_1 rounds to create d_1 red edges not in $E\big(\bigcup_{i\in[d_1]} \bar{F}'_i\big)$ (but included in $\bar{F}$).

As for a path-double-rooted tree T_j, suppose that at some point Waiter forced a copy $\bar{F}'_j$ of a rooted path $P_j - \{r'_j\} := r_j v_1 v_2 \dots v_q$, which is a part of the double-rooted path P_j with one missing edge $v_q r'_j$. Let $\bar{r}_j, \bar{v}_1, \bar{v}_2, \dots, \bar{v}_q$ be the corresponding images of vertices in B. Note also that since the length of P is at least 4, we have $q \geqslant 3$. Let $\bar{r}'_j$ be the image of r'_j in U_2. Then all edges between $\bar{r}'_j$ and $V(\bar{F}'_j)$ are free. First Waiter offers the edges $\bar{v}_q\bar{v}_{q-2}$ and $\bar{v}_q\bar{v}_{q-3}$ which, again by the hypothesis of Lemma 3.14, are free. If Client chooses $\bar{v}_q\bar{v}_{q-2}$, then in the following round Waiter offers $\bar{r}'_j\bar{v}_q$ and $\bar{r}'_j\bar{v}_{q-1}$. But then, even if Client colors $\bar{r}'_j\bar{v}_q$ blue, a red path of length $v(P_j)$ between $\bar{r}_j$ and $\bar{r}'_j$ is created, namely we have a red path $\bar{r}_j\bar{v}_1 \dots \bar{v}_{q-2}\bar{v}_q\bar{v}_{q-1}\bar{r}'_j$. Similarly, if in the previous round Client chooses $\bar{v}_q\bar{v}_{q-3}$, then in the following round Waiter offers $\bar{r}'_j\bar{v}_q$ and $\bar{r}'_j\bar{v}_{q-2}$. Again, even if Client colors $\bar{r}'_j\bar{v}_q$ blue, a red path $\bar{r}_j\bar{v}_1 \dots \bar{v}_{q-3}\bar{v}_q\bar{v}_{q-1}\bar{v}_{q-2}\bar{r}'_j$ is created (see Figure 3.3). Hence, Waiter forces a red copy of $\bar{P}_j$ of the double-rooted path P_j. Waiter repeats this strategy for each tree T_j, $j = d_1 + k$ for $k \in [d_2]$, creating d_2 extra red edges (not included in $\bar{F}$).

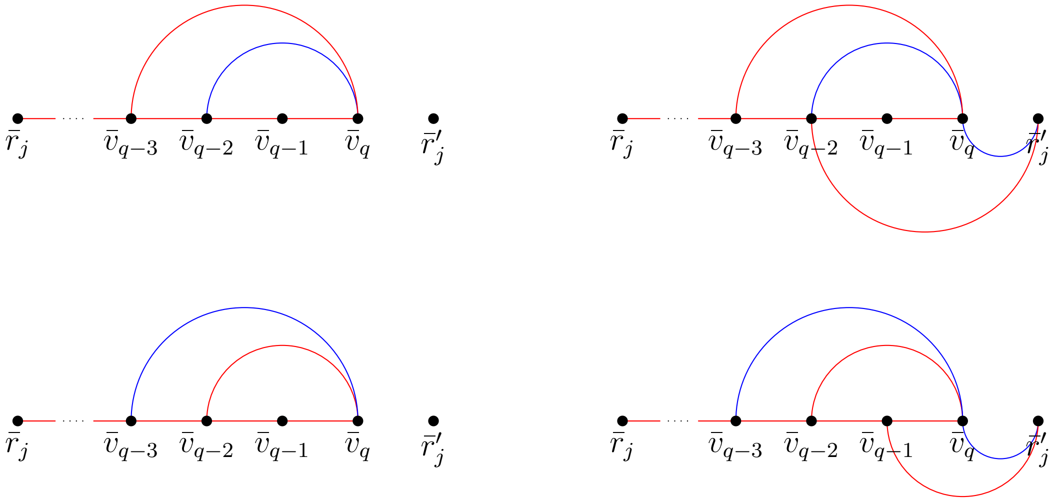

FIGURE 3.3. Connecting the roots $\bar{r}_j$ and $\bar{r}'_j$ to get a double-rooted path $\bar{P}_j$: in the first round Waiter offers edges $\bar{v}_q\bar{v}_{q-3}$ and $\bar{v}_q\bar{v}_{q-2}$.

In view of Lemma 3.14, while creating a red copy of F', Waiter creates only one red edge not included later in $\bar{F}$. Thus the above strategy guearantees the desired red copy $\bar{F}$ of F within at most $e(F) + 1 + d_2 \leqslant e(F) + d + 1$ rounds. □

While most of the previous lemmas deal with forests in which the size of each component is bounded, we will also need strategies for forests where the components can be larger but the maximum degree is bounded sufficiently.

Lemma 3.20. *Let F be a forest such that each of its components is rooted or double-rooted. Suppose that $d \geqslant 0$ of the components of F are double-rooted trees, and the two roots of such components are distant by at least 7. Let B be a complete graph on at least $v(F) + \Delta(F)$ vertices. Then Waiter can force a copy $\bar{F}$ of F in B within at most $e(F) + d$ rounds in such a way that* *(a) all roots are mapped into vertices fixed at the beginning,*

(b) every colored edge has an endpoint in a non-leaf of $\bar{F}$,

(c) every non-leaf of $\bar{F}$ is incident with at most $\Delta(F)$ blue edges whose other endpoint is in $V(B)\backslash V(\bar{F})$.

Proof. Let $T_1, T_2, \ldots, T_d$ be the components of F with two roots, and denote with x_i and y_i the roots of T_i for every $i \in [d]$. Let x_i' and y_i' be their fixed images, and let P_i be the unique path between x_i and y_i in F, where we let $d_i = e(P_i) \geqslant 7$. Before the game starts, partition $V(B)$ into vertex sets $V_1, V_2, \ldots, V_d, V_{d+1}$ such that for every $i \in [d]$, we have $|V_i| = d_i + 1$ and that V_i contains x_i' and y_i' but no other fixed image of roots in F. Let F_0 be the union of all paths P_i.

In the first stage of the game, for each $i \in [d]$, Waiter plays on each of the boards $B[V_i]$ according to the strategy from Lemma 3.2 in order to claim a Hamilton path in $B[V_i]$ between x_i' and y_i', i.e. a copy of P_i. In total, this stage lasts $\sum_{i\in[d]}(e(T_i) + 1) = e(F_0) + d$ rounds.

Afterwards, for the second stage, Waiter can extend the copy of F_0 in a greedy way to a copy of F, while keeping the images of all roots fixed. To be more precise, as long as there is a vertex x in the image of a partial embedding of F at which still some number $t \leqslant \Delta(F)$ of edges need to be claimed, Waiter simply plays t rounds in each of which she offers two edges between x and the free vertices. Note that this is possible since $v(B) \geqslant v(F) + \Delta(F)$. Moreover, this second stage lasts precisely $e(F_0) - e(F_1)$ rounds, so that after a total of $e(F) + d$ rounds Waiter has forced a copy of F.

Property (a) is immediate. Property (b) holds since in the first stage no leaf is embedded and since in the second stage, Waiter only offers edges incident to some vertex x which already belongs to the embedding and still needs at least one more neighbor, thus cannot be a leaf. Moreover, as Waiter only offers edges between such a vertex x and the remaining free vertices as long as x is still missing some neighbors, which happens at most $\Delta(F)$ rounds, property (c) follows. □

The following lemma is a variant of Theorem 5.2 in [5] and can be proven analogously. It roughly states that a spanning tree of not too large maximum degree can be forced by Waiter fast, even if the images of $r \in \{1, 2\}$ vertices are fixed at the beginning of the game. In fact, the case $r = 1$ is already covered by Theorem 5.2 in [5], while the case $r = 2$ only requires a few modifications. We therefore omit the details here, and move the proof to the appendix.

Lemma 3.21. *There exists $\alpha > 0$ and n_α such that for every $n \geqslant n_\alpha$ the following holds. Let T be tree on n vertices, with $r \in \{1, 2\}$ roots and with $\Delta(T) \leqslant \alpha\sqrt{n}$, and such that its roots (if there are two of them) are distant by at least 7. Then Waiter can force a copy $\bar{T}$ of T in K_n within at most $e(T) + r$ rounds, in such a way that all roots are mapped into vertices fixed at the beginning.*

Finally, we can conclude the following theorem.

Theorem 3.22. *There exists $\alpha > 0$ and n_α such that for every $n \geqslant n_\alpha$ the following holds. Let F be a forest on n vertices, with $\Delta(F) \leqslant \alpha\sqrt{n}$ and such that every component is non-trivial and rooted or double-rooted. Suppose that $d \geqslant 0$ of the components are double-rooted trees and for every double-rooted component its roots are distant by at least 7. Then Waiter can force a copy $\bar{F}$ of F in*

K_n within at most $e(F)+d+1$ rounds, in such a way that all roots are mapped into vertices fixed at the beginning.

Proof. Let α' be the constant promised by Lemma 3.21, and set $\alpha = \frac{\alpha'}{3}$. Whenever needed, assume that n is large enough. Note that $d \leqslant \frac{n}{8}$, since every double-rooted component is required to have at least 8 vertices. If every component of F has less than $\frac{e(F)-d}{3}$ edges, then the assertion follows from Lemma 3.19. Hence, assume from now on that there is a component T in F which has at least $\frac{e(F)-d}{3} \geqslant \frac{n}{8}$ edges and put $F' = F - V(T)$. Let $R \subseteq V(T)$ be the set of roots of T, and note that

$$n - |R| \geqslant v(F') + v(T) - 2 \geqslant v(F') + \frac{n}{8} - 2 \geqslant v(F') + \Delta(F')$$

for n big enough. Let $\bar{R}$ be the vertices in K_n that the roots in R need to be mapped to. Lemma 3.20 implies that Waiter can force a copy $\bar{F}'$ of F' in $K_n - \bar{R}$ such that the properties (a) and (b) from this lemma hold. In particular, all roots of F' are mapped to the vertices fixed at the beginning, and no edge in in $V(K_n)\backslash V(\bar{F}')$ is colored yet. Moreover, it takes her at most $e(F')+d'$ rounds, where $d' \in \{d-1, d\}$ is the number of double-rooted trees of F'. Afterwards, Waiter plays on the board $K_n - V(\bar{F}')$. In view of Lemma 3.21 she can force a copy of T, within at most $e(T) + |R|$ rounds such that the vertices of R are mapped to the fixed vertices in $\bar{R}$. For this, note that $\Delta(T) \leqslant \alpha\sqrt{n} < \alpha'\sqrt{v(T)}$. In particular, Waiter forces a copy of F as required within $e(F') + d' + e(T) + |R| = e(F) + d + 1$ rounds. □

4. Forcing spanning trees with linear maximum degree

In this section we prove Theorem 1.2.

Let $\varepsilon \in \left(0, \frac{1}{3}\right)$ be given, set $\delta = \frac{\varepsilon}{3}$, let α be the constant from Theorem 3.22, and let $b = 20\varepsilon^{-1}\alpha^{-1}$. Whenever needed assume that n is large enough. Let T be a tree on n vertices with $\Delta(T) \leqslant \left(\frac{1}{3} - \varepsilon\right) n$. We set $D = \frac{\varepsilon\alpha\sqrt{n}}{5}$, and we say that a vertex of T is *big* if its degree in T is at least D. We denote the set of all big vertices by Big_T, and we let T' be the smallest subtree of T containing all vertices from Big_T. We say that a bare path in T' is *special* if each of its inner vertices does not belong to Big_T. Let $\mathcal{P}$ be the family of all maximal special bare paths of T' whose length is at least 11, and let $\mathcal{P}'$ be the family of all the paths of $\mathcal{P}$ without their endpoints. Note that then the paths in $\mathcal{P}'$ are vertex-disjoint and have length at least 9. With I' we denote the set of all inner vertices of the paths in $\mathcal{P}'$. Moreover, we set

$$F_1 := \begin{cases} T', & \text{if } v(T') \leqslant \frac{\varepsilon n}{4}, \\ T' - I', & \text{if } v(T') > \frac{\varepsilon n}{4}. \end{cases}$$

In the following, we first give a brief overview of Waiter's strategy for a game played on $B := K_n = (V, E)$, and afterwards we provide more details on why Waiter can play as suggested and force a copy of T as required.

Strategy description. Waiter's strategy consists of three stages. Details of each stage will be given in the strategy discussion.

Stage I.
By an application of Lemma 3.18, Waiter forces a red copy $\bar{F_1}$ of F_1 within $e(F_1)$ rounds such that immediately afterwards the following holds:

(I.1) every colored edge has an endpoint in $V(\bar{F_1})$.
(I.2) all blue edges in $E_B(V(\bar{F_1}), V\backslash V(\bar{F_1}))$ form a star forest such that every star is centered at a vertex $u \in V(\bar{F_1})$ and has at most $\deg_{\bar{F_1}}(u)$ edges.

Next, let $f_1 : V(F_1) \to V(\bar{F_1})$ denote the embedding of F_1 into the red graph. Then Waiter proceeds with Stage II.

Stage II.
Let $C_1, \ldots, C_s$ be the non-trivial components of $F_2' := T\backslash E(F_1)$. For each $i \in [s]$, let R_i denote the set of vertices of C_i that belong to $V(F_1)$, and call these vertices the roots of C_i. Note that $1 \leqslant |R_i| \leqslant 2$ for every $i \in [s]$. Furthermore, C_i has two roots r_i, r_i' if and only if these vertices are connected in C_i by a path from $\mathcal{P}'$. Let $R := \bigcup_{i\in[s]} R_i$, $B_2 := B[(V\backslash V(\bar{F_1})) \cup f_1(R)]$ and $t := n - v(F_1)$. We distinguish two cases.

Case 1. If $F_1 = T'$ and all components C_i have size at most $\left(\frac{1}{3} - \delta\right) t$, then Waiter finishes the embedding of T within $e(F_2') + 1$ further rounds, by an application of Theorem 3.17 on the board B_2, and she skips Stage III.

Case 2. Otherwise, let $F_2 \subset F_2'$ be the star forest consisting of all edges incident with R in F_2'. (This includes the first and last edge of each path in $\mathcal{P}'$.) Let us assume that the vertices in R are the roots of F_2. Then, by an application of Theorem 3.17, playing on B_2 for $e(F_2)$ rounds, Waiter forces a red copy $\bar{F_2}$ of F_2 such that $f_1(R)$ is the set of roots of $\bar{F_2}$ and the following holds.

(II.1) For each $r \in R$, the star with center $\bar{r} := f_1(r)$ has size $\deg_{F_2}(r)$.
(II.2) All colored edges from Stage II intersect $V(\bar{F_2})$.
(II.3) Every vertex in $V\backslash(V(\bar{F_1}) \cup V(\bar{F_2}))$ is incident with at most one colored edge and the other endpoint of such an edge is not a leaf of $\bar{F_2}$.

Afterwards, let $f_2 : V(F_1 \cup F_2) \to V(\bar{F_1} \cup \bar{F_2})$ denote the obtained embedding (an extension of f_1) of $F_1 \cup F_2$ into Client's graph. Then, Waiter proceeds with Stage III.

Stage III.
When Waiter enters this stage, Client's graph contains a copy of $F_1 \cup F_2$. Let $F_3 := T\backslash(E(F_1) \cup E(F_2))$. Then, by an application of Theorem 3.22, within at most $e(F_3) + \frac{4n}{D} + 1$ rounds, Waiter finishes a copy of T.

Strategy discussion. We now explain, separately for each stage, why Waiter is able to play as described above. The following properties of $\mathcal{P}'$, I' and F_1 will be useful in the further analysis.

Claim 4.1. *For large enough n we have the following.*

(a) $|\mathcal{P}'| < \frac{4n}{D}$.
(b) If $v(T') > \frac{\varepsilon n}{4}$, *then* $|I'| > v(T') - \frac{52n}{D}$.

(c) $v(F_1) \leqslant \frac{\varepsilon n}{4}$.

Proof. Note that by the definition of Big_T we have $|\mathrm{Big}_T| < \frac{2n}{D}$. Clearly $\Delta(T') \leqslant |L(T')| < \frac{2n}{D}$, since every leaf of T' is a big vertex of T. Define a set $\mathrm{Good}_T = \{v \in V(T') \setminus \mathrm{Big}_T : \deg_{T'}(v) = 2\}$. Next, observe that the number of vertices of T' which belong to Big_T or have degree at least 3 in T' is less than $|\mathrm{Big}_T| + |L(T')| < \frac{4n}{D}$. Hence, $|\mathrm{Good}_T| > v(T') - \frac{4n}{D}$. Since every endpoint of a maximal special bare path is either a big vertex (including all leaves of T') or it has degree more than 2 in T', the number of maximal special bare paths in T' is less than $\frac{4n}{D}$. Thus $|\mathcal{P}'| \leqslant |\mathcal{P}| < \frac{4n}{D}$, and the first part of the claim follows.

For the second part, we assume that $v(T') > \frac{\varepsilon n}{4}$ and we denote by I the set of all inner vertices of the paths in $\mathcal{P}$. We already verified that $|\mathcal{P}| < \frac{4n}{D}$, hence the number of inner vertices of maximal special bare paths of length at most 10 is less than $\frac{40n}{D}$. Therefore

$$|I'| = |I| - 2|\mathcal{P}| > |\mathrm{Good}_T| - \frac{40n}{D} - \frac{8n}{D} > v(T') - \frac{52n}{D}.$$

In the last inequality we used the fact $|\mathrm{Good}_T| > v(T') - \frac{4n}{D}$ proved above.

In order to prove the third part of the claim, we assume that $v(T') > \frac{\varepsilon n}{4}$ and $F_1 = T' - I'$, since otherwise $v(F_1) = v(T') \leqslant \frac{\varepsilon n}{4}$. Then, based on (b), we infer that $v(F_1) = v(T') - |I'| < \frac{52n}{D} < \frac{\varepsilon n}{4}$ for large enough n. □

Stage I analysis. In view of Claim 4.1(c), we know that $v(F_1) + e(F_1) < \frac{\varepsilon n}{2} < n$. Thus, by Lemma 3.18 (with $B := K_n$), Waiter can force a red copy of F_1 as described.

Stage II analysis. When Waiter enters Stage II, she has already forced a red copy $\bar{F_1}$ of F_1, and the roots of the non-trivial components C_i of F_2' have fixed images $f_1(R)$ on the board B_2.

We start with the discussion of Case 1. For each component C_i we have $e(C_i) \leqslant \left(\frac{1}{3} - \delta\right) t$ and $|R_i| = 1$, i.e. there is exactly one root r_i, since $F_1 = T'$. Now, we put $F_2 = \bigcup_{i \in [s]} C_i$ and consider the game played on B_2, which is a complete graph on $t + s$ vertices, with some edges colored. Note that $t = e(F_2)$ and the properties (I.1) and (I.2) imply that the blue edges in B_2 form a star forest with $f_1(R)$ being the set of centers of the stars. Moreover, the number of these edges is bounded by the number of previous rounds, which is $e(F_1) \leqslant \frac{\varepsilon n}{4} < \delta t$. Thus, by Theorem 3.17 with $q = 0$, Waiter can force a red copy $\bar{F_2}$ of F_2 in B_2 within $e(F_2) + 1$ rounds and such that all vertices in R are mapped to their fixed images under f_1. Then, the union of $\bar{F_1}$ and $\bar{F_2}$ is a copy of T obtained within a total of $e(F_1) + e(F_2) + 1 = e(T) + 1$ rounds.

Next, we consider Case 2. First we will verify that

$$v(B_2) - v(F_2) > \frac{\varepsilon n}{5}. \tag{4.1}$$

If $F_1 \neq T'$, then $v(T') > \frac{\varepsilon n}{4}$, so by Claim 4.1(c) we get

$$|I'| > v(T') - \frac{52n}{D} > \frac{\varepsilon n}{4} - \frac{52n}{D} > \frac{\varepsilon n}{5}$$

for large enough n. Thus, in this case (4.1) holds, since $v(B_2) - v(F_2) \geqslant |I'|$.

If otherwise $F_1 = T'$, but there is a component C_i such that $e(C_i) > \left(\frac{1}{3} - \delta\right) t$, then for the unique root r_i of C_i we have

$$e(C_i) - \deg_{F_2}(r_i) > \left(\frac{1}{3} - \delta\right) t - \left(\frac{1}{3} - \varepsilon\right) n = \left(\frac{1}{3} - \delta\right) (n - v(F_1)) - \left(\frac{1}{3} - \varepsilon\right) n > \frac{\varepsilon n}{5},$$

where in the last inequality we put $\delta := \frac{\varepsilon}{3}$ and use Claim 4.1(c). Since $v(B_2) - v(F_2) \geqslant e(C_i) - \deg_{F_2}(r_i)$, we obtain (4.1) again.

Observe that each of the components of the rooted star forest F_2 has size at most $\Delta(T) \leqslant \left(\frac{1}{3} - \varepsilon\right) n < \left(\frac{1}{3} - \delta\right) t$, and has only one root. Therefore, by the analogous argument as in Case 1, using Theorem 3.17, Waiter can force a red copy $\bar{F}_2$ of F_2 on B_2 such that all roots are mapped to their fixed images under f_1. As $v(B_2) > v(F_2)$, Waiter needs only $e(F_2)$ rounds of Stage II. The properties (II.1) – (II.3) follow from the properties (a) – (d) in Theorem 3.17.

Stage III analysis. It follows from (4.1) and the definitions of F_3 and B_2, that $e(F_3) > \frac{\varepsilon n}{5}$. Let $C'_1, \dots, C'_{s'}$ be the non-trivial components of F_3. In view of the description of Stage I and Stage II, for every vertex $u \in V(F_1)$ we have $\deg_{\bar{F}_1 \cup \bar{F}_2}(f_2(u)) = \deg_T(u)$, so there are no vertices of F_1 in $\bigcup_{i \in [s']} C'_i$. For each $i \in [s']$ let R'_i denote the set of vertices of C'_i that belong to $V(F_2)$, and call these vertices the roots of C'_i. Note that $1 \leqslant |R'_i| \leqslant 2$ for every $i \in [s]$ and, if C'_i has two roots, then these vertices are connected in C'_i by a special bare path of length at least 7 (i.e. a path from $\mathcal{P}'$ minus its endpoints). Let $R' := \bigcup_{i \in [s]} R'_i$. Since all big vertices were embedded during Stage I, and since $e(F_3) > \frac{\varepsilon n}{5}$, we know that every vertex in F_3 has degree at most $D = \frac{\varepsilon \alpha \sqrt{n}}{5} < \alpha \sqrt{v(F_3)}$. Let d be the number of double-rooted components of F_3. Then $d \leqslant \frac{4n}{D}$, since the number of paths in $\mathcal{P}'$ is at most $\frac{4n}{D}$, as we know from Claim 4.1(a).

Consider the board $B_3 := B[(V \backslash (V(\bar{F}_1 \cup \bar{F}_2)) \cup f_2(R')]$. Because of (II.3), the vertices in R' are leaves of F_2, so by (II.2) we know that all edges in $E(B_3)$ are still free at the beginning of Stage III (except, perhaps, irrelevant edges between vertices in $f_2(R')$). Therefore we can apply Theorem 3.22, provided n is big enough. Hence Waiter can force a red copy $\bar{F}_3$ of F_3 on B_3 within at most $e(F_3) + d + 1$ rounds such that each vertex in R' is mapped to its image under f_2.

Summing up, the union of $\bar{F}_1$, $\bar{F}_2$ and $\bar{F}_3$ is a copy of T, and Waiter played at most

$$e(F_1) + e(F_2) + e(F_3) + d + 1 \leqslant n + \frac{4n}{D} \leqslant n + b\sqrt{n}$$

rounds. □

5. Avoiding trees with linear maximum degree

In the following we prove Theorem 1.3. Indeed the statement immediately follows from the result below, by considering e.g. a spanning tree consisting of two large stars connected by a bare path.

Theorem 5.1. *Let n be a large enough integer. In a Waiter-Client game on K_n, Client has a strategy to avoid two disjoint red stars of size at least $0.499n$ each.*

Proof. We will describe a randomized strategy for Client and we will show that, against any strategy of Waiter, this strategy asymptotically almost surely prevents Client from occupying two disjoint stars of size $0.499n$. Note that this is enough to prove the statement above, as it implies that Waiter does not have a strategy to always force such stars, and hence Client has a deterministic strategy as desired.

We begin by ordering the vertices $\{v_1, v_2, \dots, v_n\}$ of the board K_n arbitrarily, say, by their indices. In each round Waiter offers two free edges. For $p = 0.4$, Client's strategy is as follows:

- If Waiter offers a *cherry* $v_i v_k v_j$, that is edges $v_i v_k$ and $v_j v_k$ with $i < j$, then Client colors red the edge $v_i v_k$ with probability p, and the other edge – with probability $q = 1 - p$.
- If Waiter offers a pair of disjoint edges, that is edges $v_i v_j$ and $v_k v_\ell$ with $\{i, j\} \cap \{k, \ell\} = \varnothing$, then Client colors red any of them with equal probability.

Notice that since $p < 1/2$, in the first case Client prefers "elder" vertices, and intuitively larger red stars should appear at vertices with higher indices. Our task is to show that asymptotically almost surely at the end of the game there are no two disjoint red stars of size at least cn each, where we set $c := 0.499$.

In order to analyze large disjoint stars in C, we will focus first on two vertices, say x and y. Without loss of generality we may assume that $x < y$ in the ordering considered at the beginning of this proof. Observe that in order to get two disjoint stars in C of size at least cn each, and centered at x and y, we need to have

$$\begin{cases} \deg_C(x) \geqslant cn, \\ \deg_C(y) \geqslant cn, \\ \deg_C(x) + \deg_C(y) - |N_C(x) \cap N_C(y)| \geqslant 2cn. \end{cases} \tag{5.1}$$

Note that at the end of the game we have

$$\deg_C(x) + \deg_C(y) - |N_C(x) \cap N_C(y)| + |N_W(x) \cap N_W(y)| = n - 2,$$

and for any vertex v we have $\deg_C(v) + \deg_W(v) = n - 1$. Hence, a condition necessary for (5.1) is

$$\begin{cases} \deg_W(x) < (1 - c)n, \\ |N_W(x) \cap N_W(y)| < (1 - 2c)n. \end{cases} \tag{5.2}$$

at the end of the game.

We will show that (5.2) holds with probability $o(n^{-2})$, so that later a union bound over all pairs x, y will finish the argument. To this end we will introduce three random processes (by defining certain sets V_1, V_2, V_3), partially describing the course of the game, which enable us to show that a.a.s. the blue degree $\deg_W(x)$ or the common blue neighborhood $N_W(x) \cap N_W(y)$ is larger than what (5.2) indicates. Before we proceed, we introduce a couple of additional definitions. During the game, a vertex $v \notin \{x, y\}$ is called:

- *inactive*, if both edges xv, yv are free;
- *active*, if exactly one of the edges xv, yv has been colored so far.

In order to analyze Client's randomized strategy, we consider vertex sets V_1, V_2, V_3 while the game is proceeding, where initially all these sets are empty. We update these sets after each move, depending on the following four types of moves performed by Waiter.

(1) Suppose that in some round Waiter offers a cherry xvy (and hence v was inactive before this round), then we put v into V_1.
For the i-th move of this type, let X_i denote a random variable which indicates whether Client has colored the edge yv red or blue, that is

$$X_i = \begin{cases} 1, & yv \text{ is red } (xv \text{ is blue}); \\ 0, & yv \text{ is blue } (xv \text{ is red}). \end{cases}$$

Note that the random variables X_i are independent and $\mathbb{P}(X_i = 1) = q$ for every $i = 1, 2, \ldots$. Let $T_1 = |V_1|$ at the end of the game. Then $T_1 \leqslant n - 2$ and T_1 is a random variable which may depend on Client's moves, that is T_1 may depend on random variables $X_1, X_2, \ldots$. Furthermore, and at the end of the game

$$\deg_W(x) \geqslant \sum_{i=1}^{T_1} X_i. \tag{5.3}$$

(2) Suppose that in some round Waiter offers two edges which do not form a cherry as in (1), and such that one of them is an edge between $\{x, y\}$ and some inactive vertex v, and the other is some edge uw. We then add v to V_2 with the following restrictions. If Waiter offers two edges between $\{x, y\}$ and distinct vertices z, z', where both z and z' are inactive vertices, then we let v be any of these vertices, say z, and we let uw be the other edge, say incident with z'. In this case we add $v = z$ to V_2, but we do not add z' to V_2. Moreover, if Waiter offers two edges between $\{x, y\}$ and distinct vertices z, z', such that exactly one of z and z' is inactive, and the other is an active vertex in V_2, we treat it as a move of type 3 described below, i.e. we do not add any vertex to V_2.
Having v fixed for a move of type 2, we let Y_v denote a random variable which indicates whether Client has colored the other edge uw red or not, that is

$$Y_v = \begin{cases} 1, & uw \text{ is red}; \\ 0, & uw \text{ is blue}. \end{cases}$$

This time the random variables $(Y_v)_{v \in V_2}$ do not have to be independent, but we know that for every $v \in V_2$ we have

$$\mathbb{P}(Y_v = 1) \geqslant p.$$

(3) Suppose that in some round, Waiter offers two edges neither of type 1 nor of type 2, and such that one of them is an edge between $\{x, y\}$ and some active vertex $v \in V_2$, and the other is some edge uw. Then we add v to V_3 with the following restriction. If Waiter offers two edges between $\{x, y\}$ and distinct vertices z, z', where both z and z' are active vertices in V_2, we

then let v be any of these vertices, say z, and we let uw be the other edge, say incident with z'. In this case we add $v = z$ to V_3, but we do not add z' to V_3.

Having v fixed for a move of type 3, we let Y'_v denote a random variable which indicates whether Client has colored the other edge uw red or not, that is

$$Y'_v = \begin{cases} 1, & uw \text{ is red;} \\ 0, & uw \text{ is blue.} \end{cases}$$

Again, the random variables $(Y'_v)_{v\in V_3}$ do not have to be independent, but for every $v \in V_3$ we have

$$\mathbb{P}(Y'_v = 1) \geqslant p.$$

Moreover, the event $\{Y_v = 1, Y'_v = 1\}$ implies that xv, yv are blue or, equivalently, $v \in N_W(x) \cap N_W(y)$.

(4) Suppose that Waiter makes a move which is not of type 1, 2 or 3, then we do not update any of the sets V_1, V_2, V_3.

We can now move on to the final position analysis. Let $T_2 = |V_3|$ at the end of the game. We claim that

$$T_1 + 4T_2 \geqslant n - 2. \tag{5.4}$$

Indeed, among $n - 2 - |T_1|$ vertices v which have not been offered in a move of type 1, at most half were ignored in moves of type 2 and not included in V_2 (in case were Waiter offered two edges between $\{x, y\}$ and distinct vertices z, z' where both z and z' where inactive) and at most half of the remaining ones were ignored in moves of type 3 and not included in V_3 (in case where Waiter offered two edges between $\{x, y\}$ and distinct vertices z, z' where both z and z' were active vertices of V_2, and also in case where one of them was active and the other not).

Next, we define the following coupling. For $v \in V_3$, let D_v be a random variable such that

$$\{Y_v Y'_v = 0\} \quad \Longrightarrow \quad \{D_v = 0\},$$

and with probability distribution given by

$$D_v = \begin{cases} 1, & \text{with probability } p^2, \\ 0, & \text{with probability } 1 - p^2. \end{cases}$$

Hence

$$|N_W(x) \cap N_W(y)| \geqslant \sum_{v\in V_3} Y_v Y'_v \geqslant \sum_{v\in V_3} D_v, \tag{5.5}$$

the random variables $(D_v)_{v\in V_3}$ are independent, and they have the same probability distribution. Our task now is to estimate the probability that the events in (5.2) hold simultaneously. By a slight abuse of notation, i.e. changing the indexing of random variables $(D_v)_{v\in V_3}$ to $(D_j)_{j=1}^{T_2}$, we can bound this probability from above by

$$\mathbb{P}\left(\deg_W(x) \leqslant (1-c)n,\ |N_W(x)\cap N_W(y)| \leqslant (1-2c)n\right) \leqslant \mathbb{P}\left(\sum_{i=1}^{T_1} X_i \leqslant (1-c)n,\ \sum_{j=1}^{T_2} D_j \leqslant (1-2c)n\right). \tag{5.6}$$

Let $\mathcal{E}_1$ and $\mathcal{E}_2$ denote the events $\sum_{i=1}^{T_1} X_i \leqslant (1-c)n$ and $\sum_{j=1}^{T_2} D_j \leqslant (1-2c)n$, respectively. Next, let us define the events

$$\mathcal{F}_1 : \sum_{i=1}^{T_1} X_i \geqslant T_1 q - 2\sqrt{n\log n} \quad \text{and} \quad \mathcal{F}_2 : \sum_{j=1}^{T_2} D_j \geqslant T_2 p^2 - 2\sqrt{n\log n}.$$

Suppose that the events $\mathcal{E}_1, \mathcal{E}_2$ and $\mathcal{F}_1, \mathcal{F}_2$ occur simultaneously. Then we get that

$$T_1 q - 2\sqrt{n\log n} \leqslant (1-c)n \quad \text{and} \quad T_2 p^2 - 2\sqrt{n\log n} \leqslant (1-2c)n,$$

which in turn implies that for some positive constant $\delta < 0.0001$ and n sufficiently large we have

$$T_1 + 4T_2 \leqslant \frac{(1-c+\delta)n}{q} + \frac{4(1-2c+\delta)n}{p^2}.$$

Recall that $p = 0.4$, $q = 0.6$ and $c = 0.499$. Hence we get

$$T_1 + 4T_2 \leqslant \frac{(1-c+\delta)n}{q} + \frac{4(1-2c+\delta)n}{p^2} < 0.89n.$$

This contradicts (5.4). Therefore, if the events $\mathcal{E}_1$ and $\mathcal{E}_2$ hold, then one of the events $\mathcal{F}_1, \mathcal{F}_2$ cannot hold. By Lemma 2.1, this in turn happens with probability $o(n^{-2})$. Hence, the probability in (5.6) is bounded from above by $o(n^{-2})$. Finally, taking the union bound over all possible pairs x, y, we get that the probability that Client's graph contains two disjoint stars of size at least cn each is $o(1)$. □

6. Client-Waiter spanning tree game

6.1. **Auxiliary games.** The main goal of this subsection is to prove that Waiter can force Client to build a graph in which every small vertex set has a large common neighborhood. We start with the following lemma.

Lemma 6.1. *Let n be large enough. Let $\mathcal{H} = (X, \mathcal{F})$ be a hypergraph with $|X| = n$, $|\mathcal{F}| \leqslant n^{\log(n)}$ and such that every hyperedge contains at least $\log^3(n)$ vertices. Then in the Waiter-Client game on $\mathcal{H}$, Waiter has a strategy such that at the end of the game, the set C of Client's elements satisfies*

$$|C\cap f| \geqslant \frac{|f|}{100} \quad \textit{for every } f\in\mathcal{F}.$$

Proof. Consider the family

$$\mathcal{F}' := \left\{ f\backslash A :\ f\in\mathcal{F},\ A\subseteq f,\ |A| = \left\lfloor \frac{|f|}{100} \right\rfloor \right\}.$$

If Client claims an element in each set of $\mathcal{F}'$, then we are done. To verify that Waiter can force this, we only need to check that the condition from Theorem 2.2 holds:

$$\begin{aligned}\sum_{F\in\mathcal{F}'} 2^{-|F|} &\leqslant \sum_{\substack{k\in\mathbb{N}\\ k\geqslant \log^3(n)}} \sum_{\substack{f\in\mathcal{F}:\\ |f|=k}} \sum_{\substack{A\subset f:\\ |A|=\left\lfloor\frac{|f|}{100}\right\rfloor}} 2^{-0.99k} \leqslant \sum_{\substack{k\in\mathbb{N}\\ k\geqslant \log^3(n)}} \sum_{\substack{f\in\mathcal{F}:\\ |f|=k}} \binom{k}{0.01k} 2^{-0.99k}\\ &\leqslant \sum_{\substack{k\in\mathbb{N}\\ k\geqslant \log^3(n)}} \sum_{\substack{f\in\mathcal{F}:\\ |f|=k}} (100e)^{0.01k}2^{-0.99k} \leqslant \sum_{\substack{k\in\mathbb{N}\\ k\geqslant \log^3(n)}} \sum_{\substack{f\in\mathcal{F}:\\ |f|=k}} e^{-0.5k}\\ &\leqslant \sum_{\substack{k\in\mathbb{N}\\ k\geqslant \log^3(n)}} \sum_{\substack{f\in\mathcal{F}:\\ |f|=k}} e^{-0.5\log^3(n)} \leqslant n^{\log(n)}e^{-0.5\log^3(n)} = o(1). \qquad \square\end{aligned}$$

In the following lemma we consider a Waiter-Client game played on a graph G, i.e. the players select edges from $E(G)$. We denote by $N_G[A]$ the set of common neighbors of vertices in $A \subseteq V(G)$ in a graph G; more precisely, $N_G[A] := (\bigcap_{v\in A} N_G(v)) \setminus A$.

Lemma 6.2. *Let $\beta \in (0,1)$. Then for every large enough integer n and every $t \in \mathbb{N}$ such that $t \leqslant 0.1\log_2(n)$ the following holds. Suppose G is a graph on n vertices and for every set A of t vertices we have a set $Y_A \subset N_G[A]$ of at least βn common neighbors. Then in the Waiter-Client game on G, Waiter has a strategy such that at the end of the game, Client's graph C satisfies the following:*

$$|N_C[A] \cap Y_A| \geqslant \frac{\beta n}{200^{t+1}} \quad \textit{for every } A \subset V(G) \ \ \textit{such that } |A| = t.$$

Proof. Before the start of the game fix an orientation of the edges of G such that

$$|N^+[A] \cap Y_A| \geqslant (1-o(1))\left(\frac{1}{2}\right)^t \beta n \geqslant (\beta - o(1))n^{0.9}$$

holds for every $A \subset V(K_n)$ of size $|A| = t \leqslant 0.1\log_2(n)$, where $N^+[A]$ denotes the set of all vertices which are common outneighbors of the vertices in A. The existence of such an orientation can be proven with a simple probabilistic argument. Moreover, fix any ordering of the vertices $v_1, \ldots, v_n$. Waiter now plays the game in n stages, where in the i^{th} stage, she offers all the outgoing edges at the vertex v_i in a suitable way. Set $N_0^+[A] := N^+[A] \cap Y_A$ for every $A \subseteq V$ of size t. After Stage i, let

$$N_i^+[A] := \begin{cases} N_{i-1}^+[A], & \text{if } v_i \notin A\\ \{v \in N_{i-1}^+[A] : \ v_iv \text{ is claimed by Client in Stage } i\}, & \text{if } v_i \in A\,. \end{cases}$$

We claim that Waiter can play in such a way that for every $i \in [n] \cup \{0\}$, and after Stage i we have

$$|N_i^+[A]| \geqslant \frac{|N_0^+[A]|}{100^k} \quad \text{if} \ \ |A \cap \{v_1, \ldots, v_i\}| = k \tag{6.1}$$

for every $A \subseteq V$ of size t. To prove this, we apply induction.

The induction start ($i = 0$) is trivial. So, let $i \geqslant 1$ and assume that (6.1) is correct after Stage $i-1$ for all sets A of size t. Note that then for every A of size t we have

$$|N_{i-1}^+[A]| \geqslant \frac{|N_0^+[A]|}{100^{|A|}} \geqslant \frac{(\beta - o(1))n^{0.9}}{100^{0.1\log_2(n)}} > n^{0.1}.$$

For Stage i notice the following. Claiming edges between v_i and $N^+(v_i)$ can be represented by claiming vertices in $N^+(v_i)$; and then claiming a certain amount of elements in some set $N^+_{i-1}[A]$ represents claiming the same number of edges between v_i and $N^+_{i-1}[A]$. We thus can apply the strategy from Lemma 6.1 with the hyperedges being the sets $N^+_{i-1}[A]$ for which $v_i \in A$ holds. We claim that Property (6.1) for i follows from the lemma. Indeed, assume that $|A \cap \{v_1, \ldots, v_{i-1}\}| = k$. If $v_i \notin A$, then by definition of $N^+_i[A]$ and induction, we have $|N^+_i[A]| = |N^+_{i-1}[A]| \geqslant \frac{|N^+_0[A]|}{100^k}$ while $k = |A \cap \{v_1, \ldots, v_{i-1}, v_i\}|$. If otherwise $v_i \in A$, then $|A \cap \{v_1, \ldots, v_{i-1}, v_i\}| = k + 1$, and by induction and because of Lemma 6.1, we have $|N^+_i[A]| \geqslant \frac{|N^+_{i-1}[A]|}{100} \geqslant \frac{|N^+_0[A]|}{100^{k+1}}$. Hence, the induction is complete.

To finish the argument, note that by the end of the game we have

$$|N^+_n[A]| \geqslant \frac{|N^+_0[A]|}{100^{|A|}} \geqslant \frac{(1 - o(1)) \left(\frac{1}{2}\right)^t \beta n}{100^t} \geqslant \frac{\beta n}{200^{t+1}}$$

for every A of size t, and observe that the vertices in $N^+_n[A] \subseteq Y_A$ belong to the common neighborhood of A in Client's graph C at the end of the game. □

6.2. **Proof of Theorem 1.4.** In the following we prove Theorem 1.4 with $c := 20$. Given any large enough $n \in \mathbb{N}$, consider a tree T on n vertices with a vertex v of degree $\deg_T(v) = t := \lfloor 0.1 \log_2(n) \rfloor$ such that each vertex of T has a neighbor in $N_T(v)$, and such that each vertex in $N_T(v)$ has roughly the same degree, i.e. about $\frac{n}{0.1 \log_2(n)} < \frac{cn}{\log(n)}$. By switching the roles of the colors blue and red, and by applying Lemma 6.2 (with $\beta = \frac{1}{2}$ and $N_A := V \backslash A$ for every set A of size t), Waiter can make sure that the final graph of blue edges satisfies that every vertex set of size t has a common neighbor. But then, in the red complement there cannot be a copy of T. □

7. Concluding remarks

We studied the asymptotics of the the function $D(n)$, where $D(n)$ is the largest integer t such that for every tree T with $v(T) = n$ and $\Delta(T) \leqslant t$ Waiter has a winning strategy in $\mathrm{WC}(n, T)$. Our linear lower and upper bounds on $D(n)$ differ by a multiplicative constant. It would be interesting to now if any "proper" constant exists.

Conjecture 7.1. *There exists a constant c such that $D(n) = cn + o(n)$.*

Going further, we could ask about the above constant c, however we do not dare to guess. Theorems 1.2 and 1.3 imply that $1/3 \leqslant c < 1/2$.

7.1. **Maximum of maximum degrees of spanning trees.** A related problem is to determine $\bar{D}(n)$, where $\bar{D}(n)$ denotes the largest t with the property that there exists a tree T with $v(T) = n$ and $\Delta(T) = t$ such that Waiter has a winning strategy in $\mathrm{WC}(n, T)$. A result of Beck [2] on discrepancy games mentioned in the introduction implies that $\bar{D}(n) \leqslant n/2 + O(\sqrt{n \log n})$. On the other hand, one can show that $\bar{D}(n) \geqslant n/2 + \varepsilon\sqrt{n \log n}$ for a positive constant ε and big enough n in the following way.

The above mentioned theorem of Beck is much stronger than we state it. In fact the second term order $\sqrt{n \log n}$ is optimal. More precisely, there exists a positive constant ε' such that for every big enough t Waiter, playing on $K_{t,t}$, can force Client to build a red star of size $t/2+\varepsilon'\sqrt{t\log t}$ ([2], Theorem 18.4). Let us apply it to a spanning tree game on K_n. We begin by partitioning $V(K_n)$ into two roughly equal sets A, B. First Waiter offers the edges $E(A,B)$ of the complete bipartite graph so that she forces creating a red star of size $n/4+\varepsilon\sqrt{n\log n}$. We can assume that this star has the centre $u \in A$. Then Waiter forces greedily a red star of size $\lfloor(|A|-1)/2\rfloor$ with its centre u, by selecting edges from $E(u,A)$. In the next stage, she applies Lemma 3.2 to force a red path on the vertex set $A\backslash\{u\}$. Finally, again by Lemma 3.2, she can force a red path on the vertex set B. Thereby Waiter forces Client to build a spanning connected graph in K_n, with the maximum degree at least $n/2+\varepsilon\sqrt{n\log n}$.

Summarizing, $\bar{D}(n)$ does not differ much from the size of the greatest red star Waiter can force in K_n. We conclude that there exist positive constants $\varepsilon, \varepsilon'$ such that for every sufficiently big n we have

$$\frac{n}{2}+\varepsilon\sqrt{n\log n}\leqslant \bar{D}(n)\leqslant \frac{n}{2}+\varepsilon'\sqrt{n\log n}.$$

7.2. **Maximum degrees of spanning trees in Client-Waiter games.** We can define functions analogous to $D(n)$ and $\bar{D}(n)$, for Client-Waiter games. More precisely, let $D_{CW}(n)$ be the largest integer t such that for every tree T with $v(T)=n$ and $\Delta(T)\leqslant t$ Client can build a red copy of T in K_n. By $\bar{D}_{CW}(n)$ we denote the largest maximum degree of a spanning tree Client can build in K_n. Theorem 1.4 implies that $D_{CW}(n)=O(n/\log n)$. The key idea behind this estimation is the fact that Waiter can play so that for every vertex set A of roughly $\log n$ vertices there is a vertex connected with blue edges with all elements of A. This idea cannot be applied for sets A of order greater than $\log n$, what can be verified for example by the analogue of the Erdős-Selfridge Criterion for Client-Waiter games (where Client plays the role of Breaker). We believe that the order $n/\log n$ of our upper bound for $D_{CW}(n)$, supported also by the probabilistic intuition, is optimal.

Conjecture 7.2. *There exists a constant $\varepsilon>0$ such that $D_{CW}(n)\geqslant\varepsilon\frac{n}{\log n}$ for every sufficiently large n.*

As for $\bar{D}_{CW}(n)$, we have an upper bound $\bar{D}_{CW}(n)\leqslant n/2+O(\sqrt{n})$, due to Beck's results on discrepancy games in the Client-Waiter version (where Waiter plays int the role of Balancer; see Theorem 18.5(a) in [2]). We do not have any interesting lower bound on $\bar{D}_{CW}(n)$. We wonder if it differs much from the size of the greatest red star Client can build in K_n.

7.3. **Star forest games.** Finally, let us comment on star forest games on K_n. In Section 3 we proved that if F is a forest with $m\leqslant n$ edges such that every its component has less than $m/3$ edges, then Waiter can force a red copy of F in K_n. In Section 5 we proved that if F is a forest consisting of two stars of size, roughly, $0.499n$ each, then Client can avoid building a red copy of F in K_n. It would be interesting to study the relation between the degree sequence of stars in a star forest F and the outcome of Waiter-Client or Client-Waiter games such that a player tries to obtain a red copy of F in K_n. Lemma 3.14 is a step in this research direction.

References

1. József Beck, *Positional games and the second moment method*, Combinatorica **22** (2002), 169–216. 1
2. ______, *Combinatorial games: Tic-Tac-Toe theory*, vol. 114, Cambridge University Press Cambridge, 2008. 1, 1, 2.3, 7.1, 7.2
3. Małgorzata Bednarska-Bzdęga, *On weight function methods in Chooser–Picker games*, Theoretical Computer Science **475** (2013), 21–33. 2.2
4. B. Bollobás, *Random graphs*, no. 73, Cambridge University Press, 2001. 1
5. Dennis Clemens, Pranshu Gupta, Fabian Hamann, Alexander Haupt, Mirjana Mikalački, and Yannick Mogge, *Fast strategies in Waiter-Client games*, The Electronic Journal of Combinatorics **27** (2020), no. 3, 1–35. 1, 1, 3.1, 3.4, A, A.1, A, A, A, A
6. András Csernenszky, C. Ivett Mándity, and András Pluhár, *On Chooser-Picker positional games*, Discrete Mathematics **309** (2009), no. 16, 5141–5146. 1
7. Asaf Ferber, Dan Hefetz, and Michael Krivelevich, *Fast embedding of spanning trees in biased Maker-Breaker games*, European Journal of Combinatorics **33** (2012), no. 6, 1086–1099. A
8. Dan Hefetz, Michael Krivelevich, Miloš Stojaković, and Tibor Szabó, *Positional games*, vol. 44, Springer, 2014. 1, 2.3
9. Svante Janson, Tomasz Łuczak, and Andrzej Ruciński, *Random graphs*, John Wiley & Sons, 2011. 2.2
10. János Komlós, Gábor Sárközy, and Ednre Szemerédi, *Spanning trees in dense graphs*, Combinatorics, Probability and Computing **10** (2001), no. 5, 397—-416. 1
11. Michael Krivelevich, *Embedding spanning trees in random graphs*, SIAM Journal on Discrete Mathematics **24** (2010), no. 4, 1495–1500. 1, A
12. Douglas B. West, *Introduction to graph theory*, vol. 2, Prentice hall Upper Saddle River, 2001. 2.1

Appendix A. Forcing trees with at most two roots and bounded maximum degree

In this section we show how to modify the proof of Theorem 5.2 in [5] in order to obtain Lemma 3.21. For this, we first state a useful lemma from [5], which is a variant of Lemma 2.1 in [11], and a simple corollary of it.

Lemma A.1 (Lemma 5.1 in [5])**.** *For every $\mu' \in (0, \frac{1}{2})$ there exists $\alpha > 0$ such that the following holds for every large enough integer n. If T is a tree on n vertices such that every vertex is adjacent with at most $\alpha\sqrt{n}$ leaves in T, then T contains a leaf matching of size at least $\mu'\sqrt{n}$ or a bare path of length at least $\mu'\sqrt{n}$.*

Corollary A.2. *Let $\mu \in (0, \frac{1}{10})$. Then there exists a constant $\alpha > 0$ such that the following holds for every large enough integer n. If F is a forest without isolated vertices such that $v(F) \geqslant (1-\mu)n$ and $\Delta(F) \leqslant \alpha\sqrt{n}$ hold, then F has a leaf matching of size at least $\mu\sqrt{n}$ or a bare path of length at least $\mu\sqrt{n}$.*

Proof. Given μ, we set $\mu' = 4\mu$ and apply Lemma A.1 to obtain a constant α as output. Whenever needed, we assume that n is large enough.

Let a forest F be given as described by the statement of the lemma. Construct a tree T from F by adding one new vertex v which is made adjacent to exactly one vertex of each component of F. Note that then no leaf of T is a neighbor of v. Hence, by Lemma A.1 and since $\Delta(F) \leqslant \alpha\sqrt{n}$, T has a leaf matching of size $\mu'\sqrt{v(T)}$ or a bare path of length at least $\mu'\sqrt{v(T)}$. Note that

$$\mu'\sqrt{v(T)} \geqslant 4\mu\sqrt{(1-\mu)n} \geqslant 2\mu\sqrt{n}.$$

Hence, after removing v, there must be a leaf matching of size at least $\mu\sqrt{n}$ or a bare path of length at least $\mu\sqrt{n}$. □

Let T be a tree as given by the assumption of Lemma 3.21. In order to force a copy of the desired tree T fast, the main idea will be to first embed everything of T except from a special structure, i.e. a long bare path or a large leaf matching, in a careful greedy way, and then to finish the tree by forcing this remaining structure. Now, in case that there are $r = 2$ roots x, y with some path $P_{x,y}$ between them, we also need to make sure to claim an appropriate copy of $P_{x,y}$ between the fixed images of x, y. Depending on the structure of $P_{x,y}$, we may first create this copy of $P_{x,y}$ and then continue as in [5], or find a long bare path contained in $P_{x,y}$ which can be used as the special structure to be embedded last. Because of this, we will deal with different cases in the proof of Lemma 3.21, but in each of these cases at some point the main part of the strategy will be purely identical with the strategy presented in [5]. In the proof of Lemma 3.21, at the end of this section, we will mostly discuss our modifications and how to deal with $P_{x,y}$, while the following preparatory lemma will be used to cover those parts of the main strategy which are identical with [5].

Lemma A.3. *For every $\mu > 0$ there exists $\alpha > 0$ such that the following holds for every large enough integer n. Let a Waiter-Client game on K_n be in progress. Let T be a tree on n vertices with $\Delta(T) \leqslant \alpha\sqrt{n}$, and let $F \subset T$ be an induced subgraph of T. Assume that the following holds:*

(a) F consists of at most two components, and if the number of components is two, then in T they are connected by a bare path of length at least $\mu\sqrt{n}$.

(b) If F is a tree, then $T_0 := T \backslash E(F)$ contains at least one of the following:

(b1) a bare path of length $\mu\sqrt{n}$ which uses no vertex of F,

(b2) a matching of size $\mu\sqrt{n}$ such that each of its edges intersects $L(T)$.

(c) So far, Waiter forced a copy $\bar{F}$ of F such that every colored edge is contained in $V(\bar{F})$. Let $f : V(F) \to V(\bar{F})$ be the embedding of F into Client's graph.

Then Waiter can force a copy $\bar{T}$ of T within $e(T_0) + 1$ further rounds such that for every $v \in V(F)$ the copy of v in $\bar{T}$ is the fixed vertex $f(v)$.

Proof. Without loss of generality assume that $\mu \leqslant \frac{1}{3}$. Lemma A.3 can be proven along the lines of Theorem 5.2 in [5]. In particular, we choose $\alpha < \frac{\mu}{20}$ as in the proof of Theorem 5.2 in [5]. Moreover, we will use the notation from [5, 7], similarly to the proof of Lemma 3.20. Let $T' \subset T$ be any subgraph of T, and let $S \subseteq V(T')$ be an arbitrary set. Then an *S-partial embedding* of T' into some graph $G \subset K_n$ is an injective mapping $f : S \to V(G)$ such that $f(x)f(y) \in E(G)$ holds if $xy \in E_{T'}(S)$. We call a vertex $v \in f(S)$ *closed* with respect to T' if all the neighbors of $f^{-1}(v)$ in T' are embedded by f. If $v \in f(S)$ is not closed with respect to T', then we call it *open* with respect to T', and moreover, we write $\mathcal{O}_{T'} = \mathcal{O}_{T'}(S)$ for the set of all vertices that are open with respect to T'. The vertices not used for the embedding are collected in the set $A := V(K_n)\backslash f(S)$, and they are called *available*.

We consider two cases, depending on the number of components of F.

Case 1. Assume that F has two components. Then by (a) there is a bare path P in T of length at least $\mu\sqrt{n}$ which connects these two components.

Denote with u and w the endpoints of P, and let u_1 and w_1 be the neighbors of u and w in P, respectively. Note that $T - (V(P)\backslash\{u, w\})$ is a forest with two tree components T_1 and T_2. We let $T' \subset T$ be the forest induced by $E(T_1) \cup E(T_2) \cup \{uu_1, ww_1\}$, and note that $F \subset T'$.

The main idea of Waiter's strategy now is to (1) first extend $\bar{F}$ to a copy $\bar{T}'$ of T' without wasting any move and then (2) to obtain $\bar{T}$ by forcing an appropriate copy of $P - \{u, w\}$ while wasting at most one move. Such a strategy is already given in Case A of the proof of Theorem 5.2 in [5], and the same strategy can be applied here. During part (1) of this strategy, Waiter maintains a set S with $V(F) \subseteq S \subseteq V(T')$ as well as an S-partial embedding f of T' into Client's graph, which represents the subgraph of T' which has been forced so far. Moreover, Waiter makes sure that at any time $f(v) = g(v)$ holds for every $v \in V(F)$.

Initially, we have $S = V(F)$ and the S-partial embedding f is given by property (c). For step (1), Waiter can play the same strategy as for Stage II in Case A of the proof of Theorem 5.2 in [5], where the only modification is that $e(T') - e(F)$ rounds get played; and for step (2), she can play exactly as in the strategy of Stage III in Case A of the proof of Theorem 5.2 in [5], where only one round gets wasted. For this note that the heart of this strategy is Observation 5.3 in [5]. This observation lists two properties (i) and (ii), that can be guaranteed as long as Waiter can follow the strategy of Stage II, provided these properties were true at the beginning of this stage. Moreover, in the discussion afterwards in [5] it is shown that, as long as these properties are maintained, Waiter can follow the strategy of Stage II and then also succeed with Stage III, until a copy of T is forced as required.

It thus follows that, in order to guarantee that the same strategy can be used here, we only need to verify that the properties (i) and (ii) from Observation 5.3 in [5] hold at the beginning, i.e. when Waiter starts to extend the embedding of F towards T'. The property (i) requires that $|A| \geqslant \mu\sqrt{n} - 2$ and $e_{C\cup W}(A) = 0$ hold. The first part is satisfied at the beginning, since then $A = V(K_n)\backslash V(\bar{F})$ and hence $|A| = n - |V(F)| \geqslant |V(P)| - 2 \geqslant \mu\sqrt{n} - 2$; the second part holds because of (c). Moreover, property (ii) requires that $\deg_W(x, A) \leqslant \deg_C(x)$ for every $x \in f(S)$, which holds since by (c) there are no colored edges incident with A.

Case 2. Assume that F has exactly one component. Then by (b) we know that $T_0 = T\backslash E(F)$ contains a bare path as described in (b1) or a matching as described in (b2).

In the first case of having a bare path Waiter can play essentially the same way as in Case 1. So, we may assume from now on that in T_0 there is a matching M of size $\mu\sqrt{n}$ such that each of its edges intersects $L(T)$. In particular, there is matching $M' \subset M$ of size at least $\frac{1}{2}\mu\sqrt{n}$ such that its edges have pairwise distance at least 2. Let $L' := V(M') \cap L(T)$, set $T' = T - L'$ and note that $F \subset T'$.

The main idea of Waiter's strategy now is to (1) first extend $\bar{F}$ to a copy $\bar{T}'$ of T' without wasting any move and then (2) to obtain $\bar{T}$ by forcing an appropriate copy of M' while wasting at most one move. Such a strategy is already given in Case B of the proof of Theorem 5.2 in [5]. During part (1) of this strategy, Waiter again maintains a set S with $V(F) \subseteq S \subseteq V(T')$ as well as an S-partial embedding f of T' into Client's graph, which represents the subgraph of T' which has been forced so far. Moreover, Waiter makes sure that at any time $f(v) = g(v)$ holds for every $v \in V(F)$.

Initially, we have $S = V(F)$ and f is given by property (c). For step (1), Waiter can play exactly the same strategy as for Stage II in Case B of the proof of Theorem 5.2 in [5], where the only modification is that $e(T') - e(F)$ rounds get played; and for step (2), she can play exactly as in the strategy of Stage III in Case B of the proof of Theorem 5.2 in [5], where only one round gets wasted. For this note that the heart of this strategy is Observation 5.4 in [5] (where α is replaced with ε). This observation lists four properties (i)–(iv), that can be guaranteed as long as Waiter can follow the strategy of Stage II, provided these properties were true at the beginning of this stage. Moreover, in the discussion afterwards it is shown that, as long as these properties are maintained, Waiter can follow the strategy of Stage II and then also succeed with Stage III, until a copy of T is forced as required. The first two properties are the same as in Case 1 above, and they can be verified analogously. Properties (iii) an (iv) consider the set $S' := S \cap N_T(L')$, and they are satisfied if $\deg_W(x, f(S')) \leqslant 1$ for every $x \in A$ and $e_W(f(S'), A) \leqslant \alpha\sqrt{n}$. Both inequalities are guaranteed at the beginning, since by (c) all colored edges are contained in $V(\bar{F})$. □

Proof of Lemma 3.21. Let $\mu \in (0, \frac{1}{20})$. Apply Corollary A.2 and Lemma A.3 with input 2μ to obtain outputs $\alpha_{A.2}$ and $\alpha_{A.3}$, and let $\varepsilon > 0$ be such that Theorem 5.2 in [5] holds. We set $\alpha = \min\{\alpha_{A.2}, \alpha_{A.3}, \varepsilon, \frac{\mu}{3}\}$.

If $r = 1$ then the statement of Lemma 3.21 follows directly from Theorem 5.2 in [5]. Hence, from now on we may assume that $r = 2$. Denote the roots of T with x and y, and let P_{xy} be the unique xy-path in T. Moreover let v_x and v_y be the vertices in K_n to which x and y must be mapped, respectively. We distinguish three cases.

Case 1. Assume that at least $\mu\sqrt{n}$ inner vertices of P_{xy} have degree at least 3 in T. Then set $F := P_{xy}$ and notice that $T_0 := T \backslash E(F)$ consists of at least $\mu\sqrt{n}$ non-trivial components. In particular, T_0 must contain a matching of size $\mu\sqrt{n}$ such that each of its edges intersects $L(T)$; i.e. property (b) from Lemma A.3 holds.

In this case Waiter plays as follows: let $V_1 \subset V(K_n)$ be any subset with $v_x, v_y \in V_1$ and $|V_1| = v(P_{xy})$. Then playing the first $e(P_{xy}) + 1$ rounds on $K_n[V_1]$ according to Lemma 3.2, Waiter forces a Hamilton path H_{xy} with endpoints v_x and v_y. Afterwards, Waiter follows the strategy from Lemma A.3 for $e(T_0) + 1$ rounds, applied with $F := P_{xy}$ and $\bar{F} := H_{xy}$. This way, Waiter forces a copy of T as required.

Case 2. Assume that $7 \leqslant v(P_{xy}) < \mu n$. Then Waiter plays analogously to Case 1 with $F := P_{xy}$. In order to make sure that this is possible, we only need to argue that $T_0 := T \backslash E(F)$ satisfies property (b) from Lemma A.3.

To do so, consider first the forest $T_0' := T - V(P_{xy})$, and note that $v(T_0') > (1-\mu)n$. If T_0' contains at least $\alpha\mu n$ isolated vertices (which are leaves in T_0), then since $\Delta(T_0) \leqslant \alpha\sqrt{n}$, we find in T_0 a leaf matching of size at least $\mu\sqrt{n}$, and (b2) is satisfied. So assume otherwise, i.e. that are less than $\alpha\mu n$ isolated vertices, and let T_0'' be the forest obtained from T_0' by deleting all isolated vertices. Then $v(T_0'') > (1-2\mu)n$. Hence, by Corollary A.2 it follows that T_0'' contains a bare path P' or a leaf matching M' of size at least $2\mu\sqrt{n}$. If we find such a bare path P', then in T at most one vertex of P' can be in the neighborhood of $V(P_{xy})$. Hence, there must be a path $P \subset P'$ with $e(P) \geqslant \frac{e(P')}{2}$ which is a bare path in T, and (b1) follows. So, assume from now on that we find a leaf matching M' as described above. We then consider all the components $C_1, C_2, \ldots$ of T_0'' and denote with v_i the unique vertex in C_i which is a neighbor of $V(P_{xy})$ in T. Note that from each component C_i we can take at least one edge for a leaf matching M in T (this edge does not need to belong to M'). Moreover, if for a component C_i we have $|M' \cap E(C_i)| \geqslant 2$, then at most one edge in $M' \cap E(C_i)$ contains v_i, and hence we can take all the remaining edges in $M' \cap E(C_i)$ for a leaf matching M in T. It thus follows that there is a leaf matching M in T with $|M| \geqslant \frac{|M'|}{2} \geqslant \mu\sqrt{n}$, and thus (b2) is satisfied.

Case 3. Assume that less than $\mu\sqrt{n}$ inner vertices of P_{xy} have degree at least 3 in T, and that $v(P_{xy}) \geqslant \mu n$. Then we can split P_{xy} into at least $\mu\sqrt{n}$ edge-disjoint intervals each of length at least $\mu\sqrt{n}$, and hence, by the assumption of this subcase, at least one of these intervals must form a bare path P'. Label the endpoints of P' with x' and y'. Let F be the forest obtained from T by removing all inner vertices of P', and note that F has two components such that each of them contains one of the roots x, y. Moreover,

$$v(F) + \Delta(F) \leqslant n - (v(P') - 2) + \Delta(T) \leqslant n + 2 - \mu\sqrt{n} + \alpha\sqrt{n} < n\,.$$

Waiter now plays as follows. At first she plays according to the strategy of Lemma 3.20 (with $d := 0$ and $B := K_n$) until after $e(F)$ rounds a copy $\bar{F}$ of F is forced such that x, y are mapped to v_x, v_y, and such (b) and (c) from Lemma 3.20 hold. Let $v_{x'}$ and $v_{y'}$ be the images of x' and y'. Waiter continues as follows: in the next two rounds, Waiter first forces an edge $x'x''$ with $x'' \notin V(\bar{F})$ (by offering two such edges), and then forces an edge $y'y''$ such that $y'' \notin V(\bar{F}) \cup \{x''\}$ (by offering two such edges). This is possible by (c) from Lemma 3.20 and since $|V(K_n)\backslash V(\bar{F})| \geqslant \mu\sqrt{n} - 1 > \alpha\sqrt{n} + 4 \geqslant \Delta(T) + 4$. Afterwards, notice that by property (b) from Lemma 3.20, all edges in $V(K_n)\backslash V(\bar{F})$ are still uncolored. Now, playing only on $V(K_n)\backslash V(\bar{F})$ according to Lemma 3.2, Waiter forces a Hamilton path between x'' and y'', and finishes a copy of T as desired. □

Department of Discrete Mathematics, Faculty of Mathematics and CS, Adam Mickiewicz University,
Poznań, Poland

Email address: grzegorz.adamski@amu.edu.pl, sylwia.antoniuk@amu.edu.pl, mbed@amu.edu.pl

Institute of Mathematics, Hamburg University of Technology, Hamburg, Germany.

Email address: dennis.clemens@tuhh.de, fabian.hamann@tuhh.de, yannick.mogge@tuhh.de