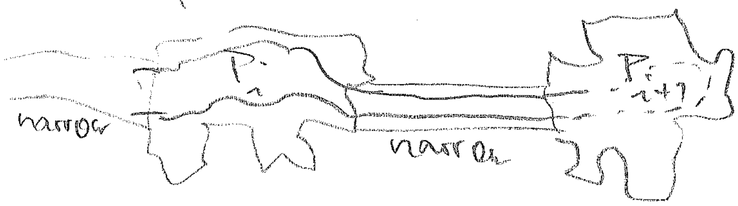
Theorem SmartDFS is $\frac{4}{3}$ - competitive.

Proof Remove from P all narrow passages $\rightsquigarrow$ polygons $P_1, \ldots$

OPT must proceed this way:



$\Rightarrow$ Can consider each P_i separately.

Claim $\forall i$: $S(P_i) \leq \frac{4}{3} C(P_i) - 2$ (THEN DONE)

Proof By induction on # split cells in Layer 1 of P_i

$\# = 0$: $S(P_i) \underset{\text{Lemma 5}}{\leq} C(P) + \frac{1}{2} E(P_i) - 5 \underset{\text{Lemma 4}}{\leq} C(P_i) + \frac{1}{2}\left(\frac{2}{3} C(P_i) + 6\right)$

$\leq \frac{4}{3} C(P_i) - 2$ ✓

$\# \geq 1$: Consider 1st split cell c in Layer 1

4 cases $\longrightarrow$ 18

P_i liegt. oder in Fall (iv) bei einfacher Adjazenz das Rechteck Q von c und c' mit $|Q| = 2$.

Analog zum Beweis von Theorem 1.15 teilen wir das Polygon P_i in zwei Teile P' und P'', ein, die beide Q enthalten. Dabei sei P'' das Polygon der Komponenten vom Typ (I) oder (II) und P' das andere.

Für $|Q| = 1$, siehe Abbildung 1.22(i) gilt offensichtlich $S(P_i) = S(P') + S(P'')$ aber auch $C(P_i) = C(P') + C(P'') - 1$. Wenden wir die Induktionsvoraussetzung auf P' und P'' an, dann erhalten wir

$$\begin{aligned} S(P_i) &= S(P') + S(P'') \\ &\leq \frac{4}{3}C(P') - 2 + \frac{4}{3}C(P'') - 2 \\ &\leq \frac{4}{3}C(P_i) + \frac{4}{3} - 4 < \frac{4}{3}C(P_i) - 2. \end{aligned}$$

Falls $|Q| = 4$ gilt sparen wir durch das Vereinigen der Polygone wieder ein paar Schritte ein. Zunächst betrachten wir P' und P'' getrennt und zählen die Schritte von c' nach c (oder umgekehrt) in beiden Polygonen. Im gesamten P_i jedoch wird der Pfad von c' nach c in P' bereits in P'' übernommen und umgekehrt. Also sparen wir hier insgesamt mindestens vier ($=|Q|$) Schritte. Dann haben wir also $S(P_i) = S(P') + S(P'') - 4$ und $C(P_i) = C(P') + C(P'') - 4$. Das ergibt mit Induktionsannahme für P' und P''

$$\begin{aligned} S(P_i) &= S(P') + S(P'') - 4 \\ &\leq \frac{4}{3}C(P') + \frac{4}{3}C(P'') - 8 \\ &= \frac{4}{3}(C(P') + C(P'') - 4) - \frac{8}{3} \\ &< \frac{4}{3}C(P_i) - 2. \end{aligned}$$

Der Fall $|Q| = 2$ ist eine leichte Übungsaufgabe, das Ergebnis gilt auch hier!

Eine optimale Strategie benötigt $\geq C(P_i)$ oder insgesamt $\geq C(P)$ Schritte, insgesamt gilt ein kompetitiver Faktor von $\frac{4}{3}$. □

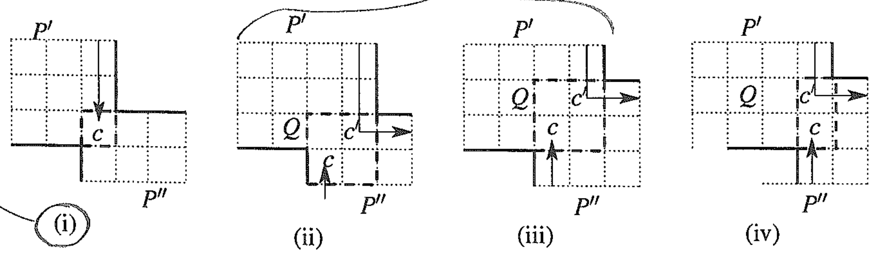
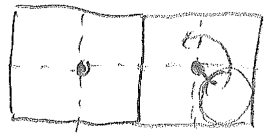


Abbildung 1.22: Ein Polygon P_i zerfällt an einer Splitzelle und wir haben Komponenten von Typ (I) oder (II). Das Quadrat/Rechteck Q liegt hier stets innerhalb von P_i.

Cleaning a cellular environment (with holes)

(Gabriely, Rimon 2000-2003)



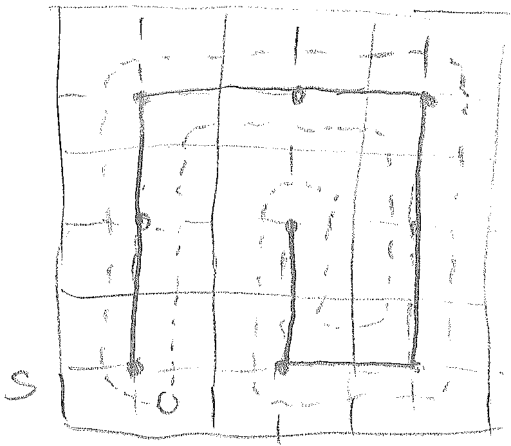
each cell has 4 subcells

robot moves between cell centers

must clean all subcells using rotating cleaning tool that cannot be lifted off the ground

easy case: all subcells free

- perform DFS on the cells
- keep cleaning tool to the right of resulting tree

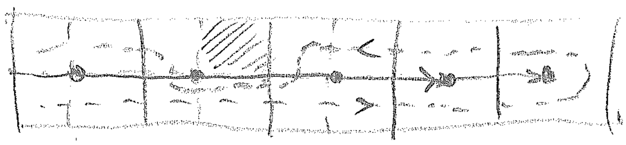


=> each subcell visited exactly once

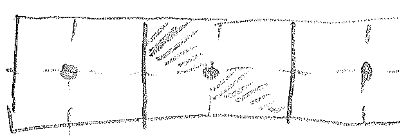
each cell twice

→ Optimal

more complicated: Some subcells may be blocked.



multiple subcell visits



no connection

needs visits from both sides

~> strategy Spiral STC (Spanning Tree Covering)

By clever counting arguments:

Let P be grid polygon with C reachable subcells
K = # free subcells diagonally adjacent to blocked su

Theorem SpiralSTC explores in time $O(C)$ all reachable subcells. The cleaning tool moves $\leq C+K$ steps.

(all edges of unit length)

Graph exploration ... visit all vertices and edges

We already know: competitive ratio = 2
(DFS + lower bound for grid polygons with holes)

(reach + topology)

... with additional constraints:

(i) tethered robot (connected by power cable to s) (= vacuum cleaner)
(ii) piecemeal exploration (robot must regularly return to s for recharging

Which task is more difficult?
Surprisingly, there is no big difference

Let $r :=$ radius of graph $G=(V,E)$
$=$ max shortest path length from s to any $p \in V$
(path length = # edges; unit weight model)

Clear: tethered robot needs cable of length $\geq r$
we assume $(1+\alpha)r$, $\alpha > 0$.
battery powered robot needs power pack capacity $2(1+\beta)r$

By clever counting arguments:

Let P be grid polygon with C reachable subcells
K = # free subcells diagonally adjacent to blocked su

Theorem SpiralSTC explores in time $O(C)$ all reachable subcells. The cleaning tool moves $\leq C+K$ steps.

(all edges of unit length)

Graph exploration ... visit all vertices and edges

We already know: competitive ratio = 2
(DFS + lower bound for grid polygons with holes)

(reach + topology) ... with additional constraints:

(i) tethered robot (Connected by power cable (= vacuum cleaner) to s)
(ii) piecemeal exploration (robot must regularly return to s for recharging

Which task is more difficult?
Surprisingly, there is no big difference

Let $r :=$ radius of graph $G=(V,E)$
= max shortest path length from s to any $p \in V$
(path length = # edges; unit weight model)

Clear: tethered robot needs cable of length $\geq r$
we assume $(1+\alpha)r$, $\alpha > 0$.
battery powered robot needs power pack capacity $2(1+\beta)r$

Lemma Let $\alpha < \beta$. Then, any $(1+\alpha)r$ strategy for tethered robots can be converted into a $2(1+\beta)r$ piecemeal exploration strategy, at $\frac{1+\beta}{\beta-\alpha}$ extra cost.

Proof Let A be a $(1+\alpha)r$ strategy for tethered robots.

Every $2(\beta-\alpha)r$ steps, A stops, at p, goes to s (to recharge), comes back to p, and resumes.

Distance from p to s: $\leq (1+\alpha)r$

$\Rightarrow$ power consumption between visits to s:

$$\leq 2(\beta-\alpha)r + 2(1+\alpha)r = 2(1+\beta)r.$$

$\Rightarrow$ have obtained $2(1+\beta)r$ peacemeal strategy.

Cost: $$\underbrace{T}_{\text{cost of } A} + \underbrace{\frac{T}{2(\beta-\alpha)r}}_{\#\text{ recharge ops}} \underbrace{2(1+\alpha)r}_{\substack{\text{max cost}\\ \text{of recharge}\\ \text{op}}} = \frac{1+\beta}{\beta-\alpha} T$$

$\square$

Conversely,

Lemma Each $(1+\beta')r$ piecemeal strategy can be converted into a $2(1+\beta')r$ tethered strategy, at $\leq$ twice the cost.

The complexities of the optimum offline solutions are unknown.

We are interested in on-line. Will consider tethered robot model, $(1+\alpha)r$ cable length, $\alpha > 0$.

Clearly, plain DFS does not work (what happens if robot runs out of cable?)

Therefore,

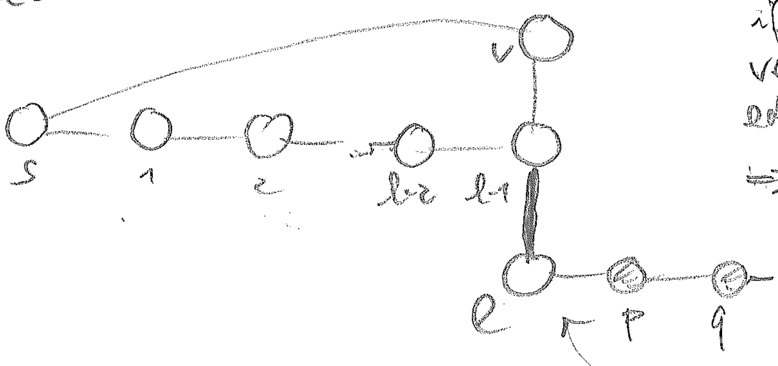
vertex — remaining cable length

```
BoundedDFS (v, l)
    if v is tagged then return;
    if l = 0 then (no more cable; must backtrack)
        if v has unexplored edge left then mark v as incomplete
                                      else mark v as explored
        return;
    mark v as tagged;
    for each unexplored edge (v,w) ∈ E
        move robot along (v,w) to w;
        mark (v,w) as explored;
        BoundedDFS (w, l-1)
        move robot back along (v,w) from w to v;
    mark v as explored
```

vertex v can be
- unexplored
- incomplete
- explored
- tagged (= on the Stack)

Looks good, but doesn't solve exploration problem!

Example:



if robot first explores vertices 1, 2, .., l, edge (l-1, l) is marked exp

$\Rightarrow$ will not be visited again, because vertex gets marked as explo when robot comes fro

CFX moves robot by DFS to this vertex, and starts Bo since it is inco

$\longrightarrow$ need better approach!

Theorem (Duncan, Kobourov, Kumar 01/06)

An unknown graph of known radius r can be explored by a tethered robot with $(1+\alpha)r$ cable within $\mathcal{O}(|E| + |V|/\alpha)$ many steps.

This strategy is $(4 + \frac{8}{\alpha})$-competitive.

2

the Thirty-Fourth Annual Symposium on Foundations of Computer Science (FOCS), 1994, pp. 2–11.

[11] A. Blum, P. Raghavan, and B. Schieber, *Navigating in unfamiliar geometric terrain*, in Proceedings of 23rd ACM Symposium on Theory of Computing (STOC), 1991, pp. 494–504.

[12] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, MIT Press/McGraw-Hill, 1990.

[13] X. Deng, T. Kameda, and C. H. Papadimitriou, *How to learn an unknown environment*, in Proceedings of the 32nd Symposium on Foundations of Computer Science (FOCS), 1991, pp. 298–303.

[14] X. Deng and C. H. Papadimitriou, *Exploring an unknown graph*, in Proceedings of the 31st Symposium on Foundations of Computer Science (FOCS), 1990, pp. 355–361. (Also in *Journal of Graph Theory*, 32(3):265–297, 1999).

[15] P. Fraigniaud, L. Gasieniec, D. Kowalski, and A. Pelc, *Collective tree exploration*, in LATIN, 2004.

[16] F. Hoffmann, C. Icking, R. Klein, and K. Kriegel, *A competitive strategy for learning a polygon*, in Proceedings of the 8th Annual Symposium on Discrete Algorithms (SODA), 1997, pp. 166–174.

[17] P. Panaite and A. Pelc, *Exploring unknown undirected graphs*, in Proceedings of the 9th Annual Symposium on Discrete Algorithms (SODA), 1998, pp. 316–322. (Also in *Journal of Algorithms*, 33:281–295,1999).

[18] C. Papadimitriou and M. Yanakakis, *Shortest paths without a map*, Theoretical Computer Science, 84 (1991), pp. 127–150.

[19] N. S. V. Rao, S. Kareti, W. Shi, and S. S. Iyengar, *Robot navigation in unknown terrains: Introductory survey of non-heuristic algorithms*, tech. rep., July 1993. ORNL/TM-12410.

[20] R. L. Rivest and R. E. Schapire, *Inference of finite automata using homing sequences*, in Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC), 1989, pp. 411–420. (Also in *Information and Computation*, 103(2):299–347, 1993).

CFX(s, r, α)
  Initially the only tree is $\{s\}$
  Repeat
    Let T_i be the closest subtree to s in G^*
    Let s_i be the closest vertex in T_i to s
    prune$(T_i, s_i, \alpha r/4, \alpha r/2)$
      producing $\mathcal{T}_i = \{T_{i_0}, T_{i_1}, \ldots T_{i_k}\}$
    // Here $T_i = \cup_{T \in \mathcal{T}_i} T$
    $\mathcal{T} \leftarrow \mathcal{T} \setminus \{T_i\} \cup \mathcal{T}_i$
    Let $T \in \mathcal{T}_i$ such that $s_i \in T$
    explore$(T, s_i, (1+\alpha)r)$
    remove all explored trees from $\mathcal{T}$
    merge any trees in $\mathcal{T}$ with common vertices
  Until $\mathcal{T} = \emptyset$

prune$(T, v, \texttt{minD}, \texttt{maxD})$
  If $T = \{s\}$, return
  Root T at v
  $\mathcal{T}_i \leftarrow \{T\}$
  for each vertex $w \in T$ such that $d_T(v, w) = \texttt{minD}$
    Let $T_w \subseteq T$ be the subtree of T rooted at w
    if $\Delta_T(v, T_w) > \texttt{maxD}$
      // Separate subtree T_w from T
      $T \leftarrow T \setminus T_w$
      $\mathcal{T}_i \leftarrow \mathcal{T}_i \cup \{T_w\}$

explore(T, s_i, l)
  Move robot from s to s_i via shortest known pat
  Move robot using depth first traversal on T
    for each incomplete vertex v visited
      Let l' be the length of rope remaining
      Call b~~DFX~~$(\cancel{v, l'})$ BoundedDFS(v, l')
      Let E' be the set of *new* explored edges
      Let V' be the set of vertices in E'
      Let T' be a spanning forest of $G' = (V', E')$
      $T \leftarrow T \cup T'$
  Move robot back from s_i to s retracing old pat

Figure 2: The CFX algorithm with start node s and rope leng $(1+\alpha)r$

CFX runs in phases

$G^* \subseteq G$: all edges and vertices explored or incomplete so f

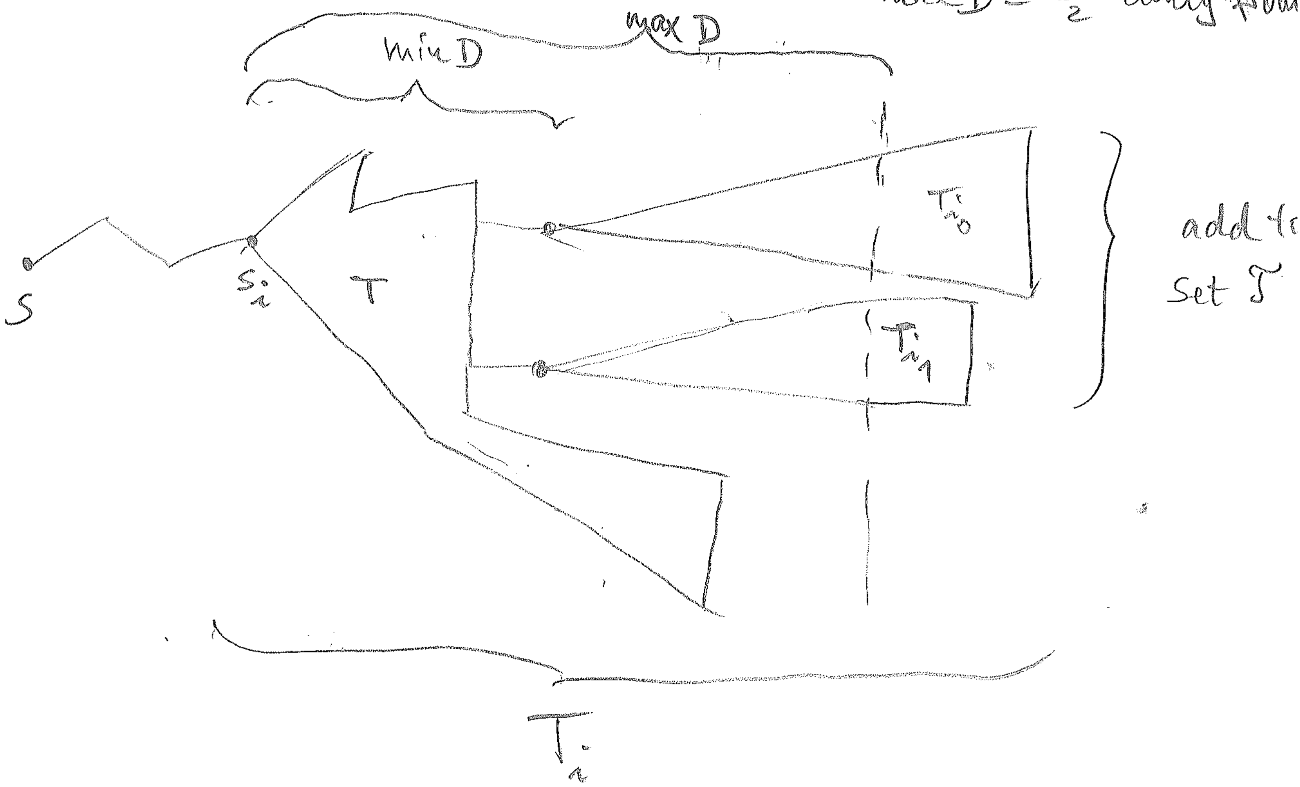
maintains set $\mathcal{T} = \{T_1, \ldots\}$ of vertex-disjoint subtrees of (
containing all incomplete vertices

single phase

$T_i \in \mathcal{T}$:= tree closest to s

$s_i \in T_i$:= vertex of T_i closest to s

prune (T_i) : roots of subtrees $minD = \frac{\alpha r}{4}$ away from
leaves of subtrees at least $maxD = \frac{\alpha r}{2}$ away from



explore (T, s_i, ℓ)

move robot from s to s_i on shortest path in G^*

perform DFS traversal on T

for each incomplete vertex in T

call BoundedDFS with remaining cable length

→ new explored edges form more trees T': add to

move robot back from s_i to s on old path

clean up $\mathcal{T}$: remove all trees explored
merge all trees sharing vertices

Initially, $\mathcal{T} = \{\{s\}\}$ (the only incomplete vertex)

robot starts with BoundedDFS$(s, (1+\alpha)r)$

Claim 1 Each incomplete vertex belongs to some $T \in \mathcal{T}$

Proof True for $\mathcal{T} = \{\{s\}\}$. Afterwards, vertices only marked unexplored in BoundedDFS.
They end up in trees T that are added to $\mathcal{T}$ (and not removed since not explored)

Claim 2 If $G^* \subsetneq G$ there exists incomplete vertex v such that $d_{G^*}(s, v) \leq r$.

$\leftarrow$ shortest path length in G^*

Proof Let v be the the incomplete vertex closest to s in G
Assume $d_{G^*}(s, v) > r$. Let $\pi = \pi_G(s, v)$ be the true closest path, by radius assumption: $|\pi| \leq$

$\pi = (s, p_1)(p_1, p_2) \dots (p_i, p_{i+1}) \dots (p_j, v)$

$\leftarrow$ first edge $\notin G^*$ (must exist, otherwise $d_{G^*}(s, v) = d_G$)

$\Rightarrow$ p_i incomplete and

$d_{G^*}(s, p_i) = d_G(s, p_i) < d_G(s, v) \leq r < d_{G^*}(s, v)$ $\lightning$ minimality of v

25.1

Claim 4 After 1st phase, each tree T in $\mathcal{T}$ is of size $|T| \geq \frac{\alpha r}{4}$.

Proof After first call BoundedDFS$(s, (1+\alpha)r)$, either G is fully explored (done!), or

phase

<u>Claim 3</u> $d_{G^*}(s, s_i) \le r$

<u>Proof</u> $\mathcal{T} \neq \emptyset \Rightarrow$ there are incomplete vertices

Let v be one closest to s $\underset{\text{Claim 2}}{\Longrightarrow}$ $d_{G^*}(s,v) \le r$

Claim 1 $\Rightarrow$ v contained in some $T' \in \mathcal{T}$

let $s' \in T'$ be closest to s in G^*

$$\Rightarrow d_{G^*}(s, s_i) \underset{\text{Def } s_i}{\le} d_{G^*}(s, s') \le d_{G^*}(s,v) \le r.$$

$\boxed{3}$

single tree T' results. Then $\mathcal{J} = \{T'\}$ and

$$|T'| \geq \text{depth}(T') \underset{\text{Def BoundedDFS}}{=} (1+\alpha)r > \frac{\alpha r}{4}.$$

Later, two types of trees are added to $\mathcal{J}$: (Later ph

① pruned parts of T_i:

bottom trees T_i^j have height $\geq \max D - \min D = \frac{\alpha r}{4}$

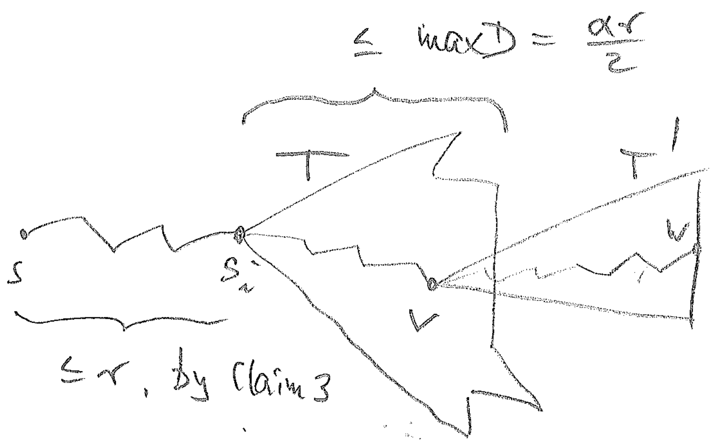
top tree T: by induction, $|T_i| \geq \frac{\alpha r}{4}$

if no pruning occurs: $|T| = |T_i| \geq \frac{\alpha r}{4}$

if pruning occurs: $|T| > \text{height}(T) \geq$

② trees T' visited in call BoundedDFS(v, ℓ')

Containing incomplete vertex w (otherwise T' complete, would be removed from



remaining cable at $v \geq (1+\alpha)r - r - \frac{\alpha r}{2} = \frac{\alpha r}{2} > 0$

$$\Rightarrow |T'| \geq |\text{height}(T')| \underset{\substack{\text{Def.}\\ \text{Bounded DFS}}}{\geq} \frac{\alpha r}{2} > \frac{\alpha r}{4}. \qquad \boxed{4}$$

Claim 5 After Explore(T, s_i, ℓ), tree T is fully explored, hence removed from $\mathcal{J}$. $\boxed{5}$

robot's Step Cost Phase 1, $\mathcal{J} = \{\{s\}\}$: $\le 2|E|$

later: only explore and BoundedDFS cause robot to move
both run on pruned tree T (= top tree of T_i)

$S_1(T)$: path length $s \dots s_i \dots s$: $\le 2r \le \frac{8|T|}{\alpha}$

$|T| \ge \frac{\alpha r}{4}$ by Claim

$S_2(T)$: DFS on T : $2|T|$

$S_3(T)$: steps of BoundedDFS calls inside explore(T,

Let $\mathcal{J}_e$ denote the set of all trees explored

$\Rightarrow \sum_{T \in \mathcal{J}_e} S_3(T) = 2|E|$, since BoundedDFS traverses only unexplored edges (but forth and b

$\Rightarrow$ # steps later $\le \sum_{T \in \mathcal{J}_e} (S_1(T) + S_2(T) + S_3(T))$

$\le \sum_{T \in \mathcal{J}_e} (\frac{8}{\alpha} + 2)|T| + 2|E|$

$= (2 + \frac{8}{\alpha}) \underbrace{\sum_{T \in \mathcal{J}_e} |T|}_{\le 2|V|} + 2|E|$

$\le 2|V|$, since each vertex can be leaf and root of a tree

$\Rightarrow$ # steps total $\le (4 + \frac{16}{\alpha})|V| + 2|E| + \underbrace{2|E|}_{\text{Phase 1}}$

$\in O(|E| + \frac{|V|}{\alpha})$.

$\le (4 + \frac{16}{\alpha}) \underbrace{|V|}_{\le |E|} + 4|E| \le (8 + \frac{16}{\alpha})|E|$.

OPT needs $2|E|$. TH

Nice extensions:

edges of different weights

→ unknown radius r

crude: guess radius r'
if too small to complete exploration:
$r' \leftarrow 2r'$
⇝ careful implementation
$O(|E| + |V| \log r)$

much better: prune$\left(T_i, s_i, \frac{\alpha d_{G^*}(s,s_i)}{4}, \frac{9\alpha d_{G^*}(s,s_i)}{16}\right)$
instead of $\mathrm{minD} = \frac{\alpha r}{4}$, $\mathrm{maxD} = \frac{\alpha r}{2}$

explore$(T_i, s_i, (1+\alpha) d_{G^*}(s, s_i))$
instead of $(1+\alpha) r$

Theorem Unknown Graph with unknown radius r can be explored in $O(|E| + \frac{|V|}{\alpha})$ steps by a tethered robot using cable of length $\leq (1+\alpha) r$.

(needs to re-prove most claims.)