

RHEINISCHE FRIEDRICH-WILHELMS-UNIVERSITÄT BONN
INSTITUT FÜR INFORMATIK I



Elmar Langetepe
Online Motion Planning

MA INF 1314

Sommersemester 2016
Manuscript: Elmar Langetepe

Altogether, the algorithm makes $(4 + \frac{8}{\alpha})|E|$ step whereas any optimal algorithm visits at least any edge once. \square

In general we assume that α is a small constant with $0 < \alpha < 1$. The above proof works for any $\alpha > 0$. The cost of the algorithm for known depth r are within $O(|E|/\alpha)$. More precisely we can show that actually $O(|E| + |V|/\alpha)$ steps are made. For this we have a closer look at the cost. bDFS work on the edges only. The DFS walk work on trees where the number of vertices is the same as the number of edges. Some of these vertices appear in two trees, so by a factor of 2 we are on the save side. The movements from s to s_i are analysed over the size of spanning trees, where vertices and edges are also the same.

The cost $K_1(T_R)$ and $K_2(T_R)$ sum up to $(2 + \frac{8}{\alpha})2|V|$.

Altogether there is an $\Theta(|E| + |V|/\alpha)$ algorithm for the exploration of arbitrary graphs.

Corollary 1.25 *The CFS-Algorithm for the constrained graph-exploration of an unknown graph with known depth has optimal exploration cost $\Theta(|E| + |V|/\alpha)$.*

Now we have some possibilities for extensions. First, we assume that the depth of the graph is unknown in the beginning. Next we would like to consider weighted edges.

1.5.1 Restricted graph-exploration with unknown depth

Let us now assume that the radius, say R , of the unknown graph G is not known. From a practical point of view, spending some cable is costly and we would like to extend the tether only if it is necessary. A first simple idea is that we guess the depth, say r , and successively double its length until the algorithm finally explores the whole graph. Obviously, the repeated application of the CFS-algorithm runs in $O(\log r|E|)$ step. As shown above we can also refine the analysis of this approach. For any bDFS step we make use of the already visited edges and directly *jump* to incomplete vertices (now with larger tether length). Therefore the bDFS steps are still subsumed by $2|E|$ steps. But we still have to take the movements to the roots of the trees into account as well as the DFS movements on the new subtrees. Therefore we have the following result.

Corollary 1.26 *Applying the CFS-Algorithmus by successively doubling the current depth r gives an algorithm that explores an unknown graph G with unknown depth R with $\Theta(|E| + (\log R)|V|/\alpha)$ steps.*

We will now show that we can get rid of the log-factor by successively adjusting r appropriately. We only exchange two calls in the main procedure. In principle, instead of the known value r we successively make use of $r := d_{G^*}(s, s_i)$, which is the smallest distance from s to one of the roots of the trees in \mathcal{T} .

More precisely, we exchange $\text{prune}(T_i, s_i, \frac{\alpha r}{4}, \frac{\alpha r}{2})$ by $\text{prune}(T_i, s_i, \frac{\alpha d_{G^*}(s, s_i)}{4}, \frac{9\alpha d_{G^*}(s, s_i)}{16})$ and $\text{explore}(\mathcal{T}, T_i, s_i, (1 + \alpha)r)$ by $\text{explore}(\mathcal{T}, T_i, s_i, (1 + \alpha)d_{G^*}(s, s_i))$. This means that the pruning-step is done with the values $\frac{\alpha d_{G^*}(s, s_i)}{4}$ and $\frac{9\alpha d_{G^*}(s, s_i)}{16}$ and the explore-step is done with tether length $(1 + \alpha)d_{G^*}(s, s_i)$.

In the beginning we have $d_{G^*}(s, s_i) = 0$, therefore we make use of some fixed constant c in the beginning and use $r := \max(d_{G^*}(s, s_i), c)$. Let $d_{G^*}(s, T)$ denote the shortest distance from s to some vertex in T inside G^* .

Lemma 1.27 *For the CFS-Algorithmus with unknown depth R we have the following properties:*

- (i) *Any incomplete vertex belongs to a tree in \mathcal{T} .*
- (ii) *There is always an incomplete vertex $v \in V^*$ with $d_{G^*}(s, v) \leq r$, until $G^* \neq G$.*
- (iii) *For the closest root s_i we have: $d_{G^*}(s, s_i) \leq r$.*
- (iv) *For all trees $T \in \mathcal{T}$ we have $|T| \geq \frac{\max(d_{G^*}(s, T), c)\alpha}{4}$. After pruning the remaining tree will be fully explored by DFS.*
- (v) *All trees ever considered in \mathcal{T} are (edge) disjoint.*

Proof. For the proof of (i),(ii),(iii) and (v) we apply the same arguments as in the proof of Lemma 1.23. It remains to show that (iv) holds. The main difference is that the size of a tree T is directly correlated to the distance from s to T , this is different from the previous argumentation.

Let us first show that the remaining tree T_i (after pruning) will be fully explored by DFS. For any vertex v in T_i we have $d_{T_i}(s_i, v) \leq \frac{9d_{G^*}(s, s_i)\alpha}{16}$, otherwise v has been cut of by pruning. Thus we have

$$(1 + \alpha)d_{G^*}(s, s_i) - d_{G^*}(s, s_i) - d_{T_i}(s_i, v) \geq \frac{7d_{G^*}(s, s_i)\alpha}{16},$$

which shows that the tether is long enough T_i will be fully explored by DFS.

By induction over the number of pruning steps we will finally show: $\forall T \in \mathcal{T} : |T| \geq \frac{\max(d_{G^*}(s, T), c)\alpha}{4}$.

In the beginning we apply bDFS from the start with tether length c . Either we explore the whole graph or we have $|T| \geq (1 + \alpha)c > \frac{\alpha c}{4}$ for the resulting spanning tree T . For simplicity we assume $d_{G^*}(s, T_i) > c$ from now on.

We would like to show that for any tree T_w , resulting from the pruning of some T_i , we have $|T_w| \geq \frac{d_{G^*}(s, T_w)\alpha}{4}$. Also the remaining tree T_i has this property.

For the remaining tree T_i (after pruning), we conclude $d_{G^*}(s, T_i) = d_{G^*}(s, s_i)$ and pruning guarantees $|T| \geq \frac{d_{G^*}(s, T)\alpha}{4}$. For a tree T_w pruned from T_i we have: $|T_w| \geq \frac{9d_{G^*}(s, s_i)\alpha}{16} - \frac{d_{G^*}(s, s_i)\alpha}{4} = 5\frac{d_{G^*}(s, s_i)\alpha}{16}$ by the pruning values. Additionally, we have $d_{G^*}(s, T_w) \leq d_{G^*}(s, s_i) + d_{G^*}(s_i, w) = (1 + \frac{\alpha}{4})d_{G^*}(s, s_i)$, since the root w of T_w is exactly $\frac{\alpha d_{G^*}(s, s_i)}{4}$ steps away from s . Für $0 < \alpha < 1$ we conclude: $d_{G^*}(s, T_w) < \frac{5d_{G^*}(s, s_i)}{4}$ and together with the above inequality we have $|T_w| > \frac{d_{G^*}(s, T_w)\alpha}{4}$.

Finally, we have to analyse the emerging spanning trees T_v , which will be constructed from the bDFS steps starting during the DFS walk in T_i . Such a tree T_v starts at some incomplete vertex v in T_i . We have $d_{G^*}(s_i, v) \leq \frac{9\alpha d_{G^*}(s, s_i)}{16}$, otherwise v would have been pruned and could not be a leaf of the rest of T_i any more. Thus we have $d_{G^*}(s, T_v) \leq d_{G^*}(s, s_i) + d_{G^*}(s_i, v) < \frac{25d_{G^*}(s, s_i)}{16}$ or $d_{G^*}(s, s_i) > \frac{16d_{G^*}(s, T_v)}{25}$. If T_v is fully explored, we are done, since the tree will be deleted. Assume that T_v still has incomplete vertices. As mentioned above we have $d_T(s_i, v) \leq \frac{9\alpha d_{G^*}(s, s_i)}{16}$. Starting from v there was a remaining tether length of $\frac{7\alpha d_{G^*}(s, s_i)}{16}$ for the construction of the incomplete T_v , which gives $|T_v| \geq \frac{7\alpha d_{G^*}(s, s_i)}{16}$. Application of $d_{G^*}(s, s_i) > \frac{16d_{G^*}(s, T_v)}{25}$ gives $|T_v| > \frac{7\alpha d_{G^*}(s, T_v)}{25} > \frac{d_{G^*}(s, T_v)\alpha}{4}$. Either we have explored everything behind v or the spanning tree T_v has size $|T_v| > \frac{d_{G^*}(s, T_v)\alpha}{4}$.

We have considered any emerging $T \in \mathcal{T}$! □

Theorem 1.28 (Duncan, Kobourov, Kumar, 2001/2006)

Applying the CFS-Algorithm with the adjustments above results in a correct restricted graph-exploration of an unknown graph with unknown depth. The algorithm is $(4 + \frac{8}{\alpha})$ -competitive. [DKK06, DKK01]

Proof. We apply the same analysis as in the proof of Theorem 1.24. For the analysis of the movements from s to the roots of the trees we make use of the correlation $|T_R| > \frac{d_{G^*}(s, T_R)\alpha}{4}$. □

For the number of steps we can also refine the analysis, analogously.

Corollary 1.29 *The above CFS-Algorithm for the restricted exploration of an unknown graph with unknown depth requires $\Theta(|E| + |V|/\alpha)$ exploration steps, which is optimal.*

Bibliography

- [AFM00] E. M. Arkin, S. P. Fekete, and J. S. B. Mitchell. Approximation algorithms for lawn mowing and milling. *Comput. Geom. Theory Appl.*, 17:25–50, 2000.
- [AKS02] Susanne Albers, Klaus Kursawe, and Sven Schuierer. Exploring unknown environments with obstacles. *Algorithmica*, 32:123–143, 2002.
- [BRS94] Margrit Betke, Ronald L. Rivest, and Mona Singh. Piecemeal learning of an unknown environment. Technical Report A.I. Memo No. 1474, Massachusetts Institute of Technology, March 1994.
- [DKK01] Christian A. Duncan, Stephen G. Kobourov, and V. S. Anil Kumar. Optimal constrained graph exploration. In *Proc. 12th ACM-SIAM Symp. Discr. Algo.*, pages 307–314, 2001.
- [DKK06] Christian A. Duncan, Stephen G. Kobourov, and V. S. Anil Kumar. Optimal constrained graph exploration. *ACM Trans. Algor.*, 2:380–402, 2006.
- [GR03] Yoav Gabriely and Elon Rimon. Competitive on-line coverage of grid environments by a mobile robot. *Comput. Geom. Theory Appl.*, 24:197–224, 2003.
- [IKKL00a] Christian Icking, Thomas Kamphans, Rolf Klein, and Elmar Langetepe. Exploring an unknown cellular environment. In *Abstracts 16th European Workshop Comput. Geom.*, pages 140–143. Ben-Gurion University of the Negev, 2000.
- [IKKL00b] Christian Icking, Thomas Kamphans, Rolf Klein, and Elmar Langetepe. Exploring an unknown cellular environment. Unpublished Manuscript, FernUniversität Hagen, 2000.
- [IKKL05] Christian Icking, Tom Kamphans, Rolf Klein, and Elmar Langetepe. Exploring simple grid polygons. In *11th Internat. Comput. Combin. Conf.*, volume 3595 of *Lecture Notes Comput. Sci.*, pages 524–533. Springer, 2005.
- [IPS82] A. Itai, C. H. Papadimitriou, and J. L. Szwarcfiter. Hamilton paths in grid graphs. *SIAM J. Comput.*, 11:676–686, 1982.
- [Lee61] C. Y. Lee. An algorithm for path connections and its application. *IRE Trans. on Electronic Computers*, EC-10:346–365, 1961.
- [Sha52] Claude E. Shannon. Presentation of a maze solving machine. In H. von Foerster, M. Mead, and H. L. Teuber, editors, *Cybernetics: Circular, Causal and Feedback Mechanisms in Biological and Social Systems, Transactions Eighth Conference, 1951*, pages 169–181, New York, 1952. Josiah Macy Jr. Foundation. Reprint in [Sha93].
- [Sha93] Claude E. Shannon. Presentation of a maze solving machine. In Neil J. A. Sloane and Aaron D. Wyner, editors, *Claude Shannon: Collected Papers*, volume PC-03319. IEEE Press, 1993.
- [Sut69] Ivan E. Sutherland. A method for solving arbitrary wall mazes by computer. *IEEE Trans. on Computers*, 18(12):1092–1097, 1969.

Index

$\dot{\cup}$	<i>see</i> disjoint union	G	
1-Layer	14	<i>Gabriely</i>	27, 29
1-Offset	14	grid-environment	8
2-Layer	14	gridpolygon	8 , 30
2-Offset	14	I	
		<i>Icking</i>	5, 18, 21
		<i>Itai</i>	8
		J	
lower bound	5	Java-Applet	18
A		K	
accumulator strategy	31	<i>Kamphans</i>	5, 18, 21
adjacent	8	<i>Klein</i>	5, 18, 21
<i>Albers</i>	30	<i>Kobourov</i>	35, 37
approximation	30	<i>Kumar</i>	35, 37
<i>Arkin</i>	30	<i>Kursawe</i>	30
B		L	
Backtrace	19	<i>Langetepe</i>	5, 18, 21
<i>Betke</i>	30	Layer	15
C		layer	27
cell	8	<i>Lee</i>	19
columns	29	Left-Hand-Rule	10–13
competitive	35, 37	Lower Bound	9
constrained	31	lower bound	8
Constraint graph-exploration	31	M	
D		<i>Mitchell</i>	30
DFS	8, 11	N	
diagonally adjacent	8 , 27	narrow passages	20
<i>Dijkstra</i>	19	NP-hard	8
disjoint union	15	O	
<i>Duncan</i>	35, 37	Offline-Strategy	5
F		Online-Strategy	5
<i>Fekete</i>	30	Online-Strategy	8

P

<i>Papadimitriou</i>	8
partially occupied cells	23
path	8
piecemeal-condition	30

Q

Queue	19
-------------	----

R

<i>Rimon</i>	27, 29
<i>Rivest</i>	30

S

<i>Schuieler</i>	30
<i>Shannon</i>	3
<i>Singh</i>	30
<i>Sleator</i>	5
SmartDFS	13, 14
spanning tree	23
Spanning-Tree-Covering	23
split-cell	14
sub-cells	23
<i>Sutherland</i>	3
<i>Szwarcfiter</i>	8

T

<i>Tarjan</i>	5
tether strategy	31
tool	23
touch sensor	8

W

Wave propagation	19
------------------------	----