Proof Section for Visibility-Based Target Tracking on Tile Graphs

Shashwata Mandal[1] and Sourabh Bhattacharya[2]

I. Tile Cover Generation

Minimal Tile Cover is the minimum tile cover given an Apriori structure till L_h. The height of the structure is the maximum level of iterations the structure can run for till no further itemsets can be generated is represented by h. The Minimal Tile Cover will be represented by MTC. The following algorithm generates the MTC.

Algorithm 1 Algorithm for Minimal Tile Cover

Input: k Height of structure, L Set of a items set till k
Output: U Minimal star tiles

1: **function** MINIMAL TILES(L,k)
2: $U \leftarrow \phi$
3: **while** $k \geq 0$ **do**
4: **for** $i \in L_k$ **do**
5: **if** $\forall x \in i, \nexists y \in U \wedge x \notin y$ **then**
6: Add i to F
7: **end if**
8: **end for**
9: Create a graph $G = (V,E)$ s.t. $V = \{v | \forall v \in F\}$ and $E = \{(u,v) | u,v \in F \wedge u \cap v \neq \phi\}$.
10: $j \leftarrow 0$
11: **while** *valid* nodes remain in G **do**
12: Add nodes with degree j to U and set linked nodes to *invalid*
13: $j \leftarrow j+1$
14: **end while**
15: $k \leftarrow k-1$
16: **end while**
17: **return** U
18: **end function**

Algorithm 1 generates the minimal tile cover on an Apriori structure. Since higher levels of the Apriori structure combine more corners compared to lower levels, the algorithm iterates from the top to the bottom of the Apriori structure. Let U be the final tile cover. U_k represents the set of tiles already added to U till level k. Similarly $C_k = \bigcup_{s \in U_k} s$ represents corners covered till level k. For each level, we find a set of disjoint tiles which maximizes $|U|$. We do this by first creating F(set of tiles excluding corners already in C_k) (Line 5) to prevent corner repetition. Then we construct a graph $G = (V,E)$ such that V is a set of tiles in F and E records intersections between those tiles (Line 9). All nodes are marked as *valid*. Nodes of G are extracted from degree 0 till $k-1$ until all *valid* nodes are gone. After each node v is extracted and added to U, all nodes such that $u \in V \wedge (u,v) \in E$ are marked *invalid*. Once level k is computed, the algorithm moves to level $k-1$.

Theorem 1. *Algorithm 1 gives a minimal tile cover.*

Proof. Since at each level we are considering tiles such that included corners are not repeated(Line 5), we are guaranteed to have a tile cover without corner repetition i.e. $\forall i,j \in U \wedge i \cap j = \phi$. Next via induction we show for each level we $|U_k|$ at each level L_k. Since h is the highest level, $U_{k+1} = \phi$ and $C_{k+1} = \phi$ where our basecase is $k = h$. Since higher level have move corners in each tile, maximizing $|U_k|$ results in a minimizing $|U|$. Let us assume that our policy of iteratively removing the lowest degree vertices in G doesn't give us the maximum $|U_k|$ even if $|U_{k+1}|$ is maximum. Let the lowest degree of vertex be m. But if we choose a vertex of degree greater than m, we end up removing more nodes which lowers the number of tiles added to U_k which is a contradiction. Hence our graph search technique of iteratively removing the lowest degree vertices is optimal. We know that the graph search maximizes $|U_k|$. So $|U_h|$ is maximum, since h is the highest level. Now we assume that $|U_k|$ is maximum for k then for $k-1$, F is generated from U_k. Since we already know the graph search technique generates the maximum U_k if $|U_{k+1}|$ is maximum. By induction we can say this hold true for any k. Hence the algorithm is optimal and U is the minimal tile cover. ■

*This work was not supported by any organization

[1]Department of Computer Science, Iowa State University, Ames, IA 50010, USA, [2]Department of Mechanical Engineering, Iowa State University, Ames, IA 50010, USA smandal@iastate.edu, sbhattac@iastate.edu