1-Complex s, t Hamiltonian Paths: Structure and Reconfiguration in Rectangular Grids

Rahnuma Islam Nishat[1], Venkatesh Srinivasan[2], and Sue Whitesides[2]

[1] Department of Computer Science, Ryerson University, Toronto, ON, Canada
[2] Department of Computer Science, University of Victoria, Victoria, BC, Canada
rnishat@ryerson.ca,{srinivas,sue}@uvic.ca

Abstract. We give a complete structure theorem for 1-complex s, t Hamiltonian paths in rectangular grid graphs. We use the structure theorem to design an algorithm to reconfigure one such path into any other in linear time, making a linear number of *switch* operations in grid cells.

1 Introduction

Let $\mathbb{G}$ be an $m \times n$ *rectangular grid graph*, which is an induced, embedded subgraph of the infinite integer grid and has m rows and n columns in an $(m-1) \times (n-1)$ rectangle $R_{\mathbb{G}}$. Let s and t be the top left and bottom right corners of $R_{\mathbb{G}}$. We define a class of s, t Hamiltonian paths (s and t are the endpoints of the Hamiltonian path) that we call *1-complex paths*. A 1-*complex path* is an s, t

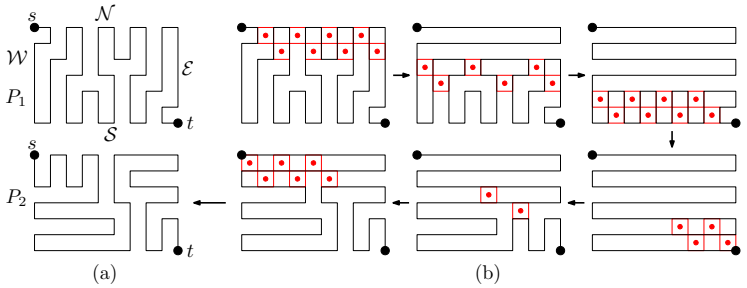

Fig. 1. (a) Two 1-complex s, t Hamiltonian paths P_1 and P_2. Sides of $R_{\mathbb{G}}$ are $\mathcal{N}, \mathcal{S}, \mathcal{W}, \mathcal{E}$. (b) A sequence of *switches* (shown by red dots) taking P_1 to P_2.

Hamiltonian path of $\mathbb{G}$, where each vertex of $\mathbb{G}$ can be connected to a node on one of the sides $\mathcal{E}, \mathcal{W}, \mathcal{N}, \mathcal{S}$ of $\mathcal{R}_{\mathbb{G}}$ by a straight line segment on the path, as in Fig. 1(a). We reconfigure such paths using *switches* in 1×1 grid *cells*. Let $R_{\mathbb{G}}$ be covered by an s, t path and 0 or more disjoint cycles (e.g., initially $R_{\mathbb{G}}$ is covered by an s, t Hamiltonian path). If a cell has two parallel grid edges that belong to the cover and two parallel grid edges that do not, then a *switch* exchanges the

edges in that cell that belong to the cover for the two edges not in the cover (see Fig. 1(b)). A switch produces a new cover comprising an s, t path and 0 or more disjoint cycles. Whether a cell may be switched depends on the cover.

The question we ask is this: given any two 1-complex paths of $\mathbb{G}$, can one of them be reconfigured to the other with only $O(|\mathbb{G}|)$ switches in grid cells, and if so, can the sequence of switches to be performed be computed efficiently?

As shown in Fig. 2, a "cross-separator" (i.e., a subpath η_i joining nodes on opposite sides of $R_{\mathbb{G}}$) may have many forms, depending on whether and where this subpath has bends. In previous work [9] we introduced a special case of 1-complex s, t Hamiltonian paths we called "simple". By definition, a simple s, t path has only straight cross-separators, so none of the subpaths in Fig. 2 that contain bent cross-separators can occur in simple paths.

A key contribution of this work is a complete structure theorem for 1-complex paths, building on the work in [9] for the special case of simple paths. Using our structure theorem for 1-complex paths, we achieve a linear time algorithm for reconfiguration of 1-complex s, t Hamiltonian paths $P_{s,t}$. Roughly, our algorithm uses the structure of a $P_{s,t}$ to define smaller sub-rectangles within $R_{\mathbb{G}}$. Path $P_{s,t}$ determines s', t' Hamiltonian paths $p'_{s',t'}$ within each sub-rectangle. We define the sub-rectangles so that our structure theorem as well as our new reconfiguration tools (Section 4) apply to each path $p'_{s',t'}$. See Fig. 2 and Section 5.

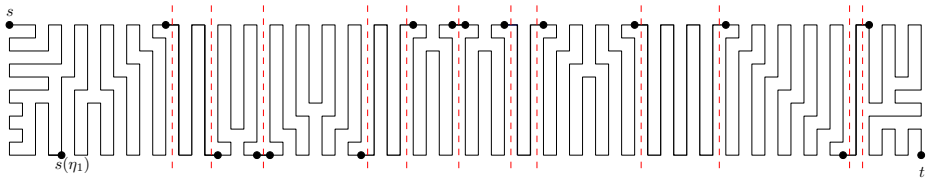

Fig. 2. The structure of the s, t path is used to break $\mathbb{G}$ into sub-rectangles.

The existence problem for Hamiltonian paths in rectangular grids [5] and non-rectangular grids ([2], [7]) has long been studied, as well as enumeration [6] and generating functions [1] for such paths. In recent years, reconfiguration problems for Hamiltonian cycles ([13], [11], [10]) and for Hamiltonian paths [9] have attracted attention. Here we advance knowledge on structure and reconfiguration of Hamiltonian paths, motivated by the many applications of both Hamiltonian cycles and Hamiltonian paths in grid graphs (see e.g. [8], [4], [12], [3]).

Our contributions:

(1) a complete structure for 1-complex s, t Hamiltonian paths (Section 3);

(2) two powerful new reconfiguration tools that find cells to switch and then sequence the switches to create straight path segments (Section 4); and

(3) an algorithm to reconfigure any 1-complex Hamiltonian path to any other such path in $O(|\mathbb{G}|)$ time, making $O(|\mathbb{G}|)$ switches in 1×1 grid *cells*, where $|\mathbb{G}|$ is the size of the grid graph $\mathbb{G}$ (Sections 5 and 6).

2 Preliminaries

We define terminology and an assumption used throughout the paper. These have previously been defined in [9], but we repeat them here for completeness.

Assumption. $\mathbb{G}$ is an $m \times n$ grid graph, where $m, n \geq 4$, and α and β are the bottom left and top right corner vertices of $\mathbb{G}$. Without loss, *we assume the input* 1*-complex path* $P_{s,t}$ *visits* α *before* β. The target 1-complex path for the reconfiguration as well as intermediate configurations may visit β before α.

A vertex of $\mathbb{G}$ with coordinates (x, y) is denoted by $v_{x,y}$, where $0 \leq x \leq n-1$ and $0 \leq y \leq m-1$. The top left corner vertex s of $\mathbb{G}$ is $(0, 0)$; the positive x-direction is rightward and the positive y-direction is downward. We use the terms *node* and *vertex* interchangeably.

Column x of $\mathbb{G}$ is the shortest path of $\mathbb{G}$ between $v_{x,0}$ and $v_{x,m-1}$, and *Row* y is the shortest path between $v_{0,y}$ and $v_{n-1,y}$. We call Columns 0 and $n-1$ the *west* ($\mathcal{W}$) and *east* ($\mathcal{E}$) boundaries of $\mathbb{G}$ (i.e., sides of $R_{\mathbb{G}}$), respectively, and Rows 0 and $m-1$ the *north* ($\mathcal{N}$) and *south* ($\mathcal{S}$) boundaries (sides).

Throughout this paper, a *1-complex path* means a 1-complex s, t Hamiltonian path P of $\mathbb{G}$; P visits each node of $\mathbb{G}$ exactly once and uses only edges in $\mathbb{G}$. We denote by $P_{u,v}$ the directed subpath of P from u to v. Straight subpaths of P are called segments, denoted $seg[u, v]$, where u and v are the segment endpoints.

Cookies and Separators. Every internal node v of $\mathbb{G}$ lies on an *internal subpath* of P, namely the subpath joining the first boundary vertices v_s and v_t met when travelling along P from v toward s and toward t, respectively. Such an internal subpath is called a *cookie* if v_s and v_t lie on the same boundary, a *corner separator* if they lie on adjacent boundaries, and a *cross-separator* if they lie on opposite boundaries. Because P is 1-complex, a cross-separator can either be *bent* and have two adjacent bends, or it can be *straight*.

A cookie c has one of four types, $\mathcal{N}$, $\mathcal{S}$, $\mathcal{E}$, and $\mathcal{W}$, depending on the boundary where c has its *base*. A cookie consists of three segments of P. The common length of the two parallel segments of c measures the *size* of c.

We say a corner separator *cuts off* a corner of $\mathbb{G}$. Traveling along $P_{s,t}$, we denote the i-th corner separator cutting off s by μ_i, where $0 \leq i \leq j$. We denote its internal bend by $b(\mu_i)$, and its endpoints by $s(\mu_i)$ and $t(\mu_i)$, where $s(\mu_i)$ is the first endpoint met. We use similar notation for the i-th corner separator ν_i cutting off t. A corner separator that has one of its endpoints connected to s or t by a segment of P is called a *corner cookie*. P can have at most two corner cookies, one at either end.

Similar to the corner separators, we denote the i-th cross-separator met along $P_{s,t}$ by η_i, and its endpoints by $s(\eta_i)$ and $t(\eta_i)$, where $s(\eta_i)$ is the first endpoint met. If η_i is bent, then $b(\eta_i)$ denotes the first bend met. In total, we have j corner separators μ_i cutting off s, and k cross-separators η_i, and ℓ corner separators ν_i cutting off t.

Runs of Cookies. A *run of cookies* is a subpath of P consisting of cookies of the same type, spaced one unit apart and joined by the single boundary edges between them, possibly extended at either end by an edge joining a cookie

endpoint to an adjacent boundary vertex. A run of cookies is denoted $Run[u, v]$, where u and v belong to the same boundary and delimit the range of boundary vertices covered; $Run[u, v]$ may consist of a single boundary edge (u, v).

To describe the path structure, we define three types of runs, depending on the cookie sizes along the run: the sizes may remain the same, or be non-increasing (denoted $Run^{\geq}[u, v]$) or non-decreasing ($Run^{\leq}[u, v]$). Runs are assumed to have cookies of the same size unless specified otherwise.

Canonical Paths. A *canonical path* $\mathbb{P}$ is a 1-complex path with no bends at internal vertices. If m is odd, $\mathbb{P}$ can be type $\mathcal{E}$-$\mathcal{W}$, filling rows of $\mathbb{G}$ one by one; if n is odd, $\mathbb{P}$ can be of type $\mathcal{N}$-$\mathcal{S}$, filling columns. There are no other types.

Observation 1. [9] For P to be Hamiltonian, the subpath $P_{s,\alpha}$ of $P_{s,t}$ must cover all the $\mathcal{W}$ vertices, no corner separators can cut off α or β, and $P_{s,t}$ must visit β before visiting any other $\mathcal{E}$ vertices. It follows that the corner separators μ_i cutting off s must occur in $P_{s,t}$ before α, and that the corner separators ν_i cutting off t occur in $P_{s,t}$ after β. Furthermore, all cross-separators η_i occur on $P_{s,t}$ between α and β, and the number $k \geq 1$ of them must be odd. □

3 Structure of 1-Complex Path P of $\mathbb{G}$

We regard P in its directed form $P_{s,t}$ as composed of an *initial* subpath $P_{s,s(\eta_1)}$, followed by a *middle* subpath $P_{s(\eta_1),t(\eta_k)}$, and a *final* subpath $P_{t(\eta_k),t}$. By reversing the edge directions, the final subpath of $P_{s,t}$ can be viewed as the initial subpath of the path $P_{t,s}$ from t to s. The grid $\mathbb{G}$ can be rotated by π to place t in the upper left corner and s in the lower right corner. Thus, apart from changes in notation, the structural possibilities for the final and initial subpaths of $P_{s,t}$ are the same, and we do not discuss the final subpath further.

We present a series of observations whose straightforward proofs may require some reflection. They lead to a *structure theorem* at the end of this section.

Initial subpath The structure of the initial subpath depends on the form of the first cross-separator η_1 of $P_{s,t}$. As η_1 may be bent or straight, there are five possible forms for it (in Fig. 3(a), see forms A, B, D, F, and G). We discuss and give the structure of the initial subpath for each of these five forms.

In the observations below, w is the vertex on $\mathcal{N}$ just one column west of $t(\eta_1)$, and a' is the vertex on $\mathcal{S}$ in the same column as w when η_1 has form G, and one column west of $s(\eta_1)$ otherwise. The vertex $v_{0,2}$ is denoted by a.

Separator η_1 has form B or G. The initial subpath has the same structure as the initial subpath of a *simple* s, t *path* described in [9].

Observation 2 (η_1 form B or G). (a) If $t(\eta_1)$ is in Column 1 next to s on the $\mathcal{N}$ boundary, then the initial subpath consists of two boundary segments $seg[s, \alpha]$ and $seg[\alpha, s(\eta_1)]$ on the $\mathcal{W}$ and $\mathcal{S}$ boundaries, respectively; see Fig. 3(b).

(b) If $j = 0$ and $x(t(\eta_1)) > 1$, $P_{s,s(\eta_1)}$ must have a corner cookie containing s and w, then next either $seg[a, \alpha]$ $Run[\alpha, a']$, or $Run[a, \alpha]$ $seg[\alpha, a']$, or $Run^{\geq}[a, \alpha]$ $seg[\alpha, u]$ $Run^{\leq}[u, a']$, where u is at least two units from α on $\mathcal{S}$ or $Run^{\geq}[a, u']$

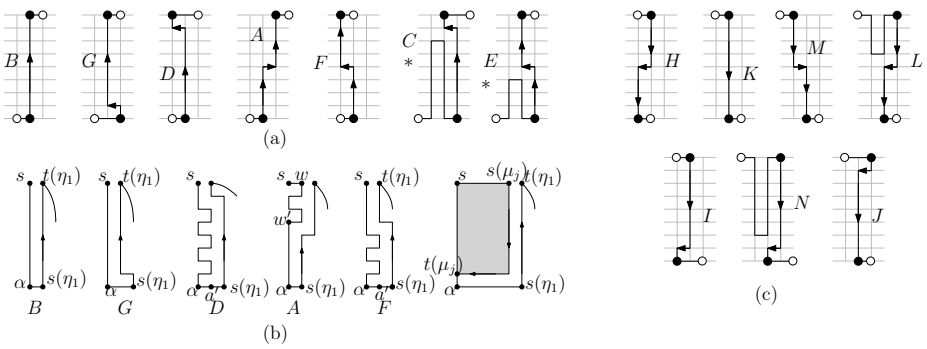

Fig. 3. (a) Seven forms for $\mathcal{S}$ to $\mathcal{N}$ cross-separators; η_1 cannot have forms E or C as any $\mathcal{S}$ cookie preceeding η_1 must be included in the initial subpath, not the middle subpath. Path P must contain the boundary segments shown. (b) Degenerate cases for different forms of η_1. (c) $\mathcal{N}$ to $\mathcal{S}$ forms for cross-separators.

$seg[u', \alpha]$ $Run^{\leq}[\alpha, a']$, where u' is at least two units from α on $\mathcal{W}$; then ending with $seg[a', s(\eta_1)]$.

(c) If $j \geq 1$, then $P_{s,t(\mu_j)}$= $Run[s, s(\mu_1)]$ μ_1 $Run[t(\mu_1), s(\mu_2)]$ μ_2 $\ldots$ μ_i $Run[t(\mu_i), s(\mu_{i+1})]$ μ_{i+1} $\ldots$ μ_{j-1} $Run[t(\mu_{j-1}), s(\mu_j)]$ μ_j. Since μ_j ends at $t(\mu_j)$ on $\mathcal{W}$, $s(\mu_1)$ must lie on $\mathcal{N}$ for j odd and on $\mathcal{W}$ for $j \geq 1$ even. Path $P_{t(\mu_j),s(\eta_1)}$ either has the structure of $P_{a,s(\eta_1)}$ in (b), or as shown in Fig. 3(b), consists of edge $(t(\mu_j), \alpha)$ followed by $seg[\alpha, s(\eta_1)]$. □

Separator η_1 has form D. The node below w must be a bend (else P cannot visit all the vertices in the column of $t(\eta_1)$); this bend must connect to $\mathcal{W}$ forming a corner cookie. Thus $j = 0$.

Observation 3 (η_1 form D). (a) If $x(t(\eta_1)) = 1$, $P_{s,s(\eta_1)} = seg[s, a]$ $Run[a, \alpha]$ $seg[\alpha, s(\eta_1)]$, where the $\mathcal{W}$ cookies have unit size. See Fig. 3(b). (b) Otherwise, $P_{s,s(\eta_1)}$ consists of a corner cookie containing s and w, followed by $P_{a,s(\eta_1)}$ with the structure given in Observation 2(b). □

Separator η_1 has form A or F. We define two rectangular regions of $\mathbb{G}$ covered by the initial subpath, and use them in designing our algorithm in Sections 5 and 6. Let w' denote the vertex on the $\mathcal{W}$ boundary in Row $y(b(\eta_1))$ for form F, and in Row $y(b(\eta_1)) - 1$ for form A. Let w'' be the vertex on the $\mathcal{W}$ boundary one row below w'. We denote by $\mathbb{R}_s$ the rectangular region of $\mathbb{G}$ that is delimited by Columns 0 and $x(w)$ and Rows 0 and $y(w')$; the rectangular region delimited by Columns 0 and $x(a')$ and Rows $y(w'')$ and $m - 1$ is denoted $\mathbb{R}_\alpha$.

Observation 4 (η_1 form A or F). (a) If $t(\eta_1)$ is in Column 1, η_1 has form F, and the initial subpath is $(\mathbb{R}_s = seg[s, w'])seg[w', w''](\mathbb{R}_\alpha = Run[w'', \alpha]$, $seg[\alpha, a'])$ $seg[a', s(\eta_1)]$. If $s(\eta_1)$ is in Column 1, η_1 has form A, and the initial subpath is $(\mathbb{R}_s = Run[s, w'])seg[w', w''](\mathbb{R}_\alpha = seg[w'', \alpha], seg[\alpha, a'])$ $seg[a', s(\eta_1)]$. In both cases, the run contains unit size $\mathcal{W}$ cookies. See Fig. 3(b).

(b) Otherwise, if $j = 0$, then $P_{s,w'}$ contains a run of $\mathcal{W}$ cookies $Run[s, w']$ of size $x(w)$. If $j \geq 1$, then $P_{s,w'} = Run[s, s(\mu_1)]\ \mu_1\ Run[t(\mu_1), s(\mu_2)]\ \mu_2\ \ldots\ \mu_i\ Run[t(\mu_i), s(\mu_{i+1})]\ \mu_{i+1}\ \ldots\ \mu_{j-1}\ Run[t(\mu_{j-1}), s(\mu_j)]\ \mu_j$, followed by $Run[t(\mu_j), w']$ if w' and $t(\mu_j)$ do not coincide. $P_{w'',s(\eta_1)}$ has the structure of $P_{a,s(\eta_1)}$ in Observation 2(b). □

We say $\mathbb{R}_s$ is $\mathcal{W}$ *compatible* if it contains an even number of rows, and $\mathbb{R}_\alpha$ is $\mathcal{W}$ *compatible* if it contains an odd number of rows. The next lemma is used to prove correctness of an algorithm in Section 5.

Lemma 1. *When η_1 has form A or F: (a) $\mathbb{R}_s$ is $\mathcal{W}$ compatible iff any $\mathcal{N}$ cookie in $\mathbb{R}_s$ has even size; $\mathbb{R}_\alpha$ is $\mathcal{W}$ compatible iff any $\mathcal{S}$ cookie in $\mathbb{R}_\alpha$ has even size. (b) At least one of $\mathbb{R}_s$ and $\mathbb{R}_\alpha$ must be $\mathcal{W}$ compatible.*

Middle subpath The middle subpath $P_{s(\eta_1),t(\eta_k)}$ by definition contains all the cross-separators. Since P is 1-complex, each internal grid point of $\mathbb{G}$ is connected to a boundary by a segment of P. Hence cross-separators are either straight, or bent toward $\mathcal{E}$ or $\mathcal{W}$ by one unit from their start nodes $s(\eta_i)$. Fig. 3(a) and (c) contain the exhaustive list of $\mathcal{S}$ to $\mathcal{N}$ and $\mathcal{N}$ to $\mathcal{S}$ cross separators, respectively.

Observation 5 (constraints on η_i, η_{i+1}). (a) $x(t(\eta_i)) = x(s(\eta_i)) \pm 1$ for bent η_i, and $x(t(\eta_i)) = x(s(\eta_i))$ for straight η_i.
(b) $x(t(\eta_{i+1})) = 1 + x(s(\eta_i))$ because the roundtrip made by $P_{s(\eta_i),t(\eta_{i+1})}$ must cover all grid points between η_i and η_{i+1}.
(c) By (a) and (b), nodes $s(\eta_{i+1})$ and $t(\eta_i)$ are 1, 2, or 3 units apart ($i < k$). Node $s(\eta_{i+1})$ lies to the right of $t(\eta_i)$, and when the nodes are 3 apart, P must have one cookie between them to cover the intervening points on the boundary. □

To describe the middle subpath structure, we introduce four special cases E, D, C, and G for form F and four special cases L, I, N, and J for form H; see Fig. 3. Note that forms H and F both have two vertical segments of length at least two. Forms E and C describe the scenario that an $\mathcal{S}$ cookie occurs between η_{i-1} and η_i. Like form F, forms D and G turn to the left (toward $\mathcal{W}$) at the first bend, but unlike F, their internal bends occur one unit from the $\mathcal{N}$ or $\mathcal{S}$ boundary. The four special cases L, I, N, and J for form H are described similarly. Which forms can appear consecutively is determined by Observation 5.

Theorem 1 (1-complex Path Structure). *Let $P_{s,t}$ be a 1-complex path with k cross-separators and j and ℓ corner separators cutting off s and t, respectively.*
initial subpath $[P_{s,s(\eta_1)}]$ *Its structure is given by Observations 2- 4.*
middle subpath $[P_{s(\eta_1),t(\eta_k)}]$ *Its structure is given by Observation 5 and the text following the observation.*
final subpath $[P_{t(\eta_k),t}]$ *As $P_{t,t(\eta_k)}$ is the initial subpath of $P_{t,s}$ (the reverse of $P_{s,t}$), the forms for the final and the initial subpaths are the same.*

4 Zip Operation

In this section, we define a *zip* operation Z (zip for short) that applies a sequence of switches to cells that appear on two sides of a line (row or column) of $\mathbb{G}$. The cells must be *switchable*, as described in Section 1. The line, called a *zipline* and denoted $l_z^{q_1,q_2}$ (the superscript may be omitted for short), is directed from node q_1 to node q_2 where q_1 and q_2 are not corners of $R_{\mathbb{G}}$.

As mentioned in Section 1, a *cycle-path cover* $\mathbb{P}$ of $\mathbb{G}$ is a set of cycles and paths that collectively cover all the vertices of $\mathbb{G}$.

The *zone* R_Z of a zip is a rectangle determined by the zipline $l_z^{q_1 q_2}$ and the two adjacent and parallel grid lines $l_a = [a_1, a_2]$ and $l_b = [b_1, b_2]$, where a_1 and b_1 are adjacent to q_1 on a boundary of $\mathbb{G}$ and a_2 and b_2 are adjacent to q_2 on the opposite boundary. Thus the corners of R_Z are a_1, a_2, b_1, b_2. We call the subgraph of $\mathbb{G}$ induced by l_a and l_z the *main track* tr of the zip, and the subgraph with sides l_z and l_b the *side track* tr'.

To describe the structure of a Hamiltonian path P inside the zone R_Z of a zip, we define the notion of a *local cookie*. See Fig. 4.

Definition 1 (local cookie). Let f' be a switchable cell for P in the side track tr' of the zone R_Z of a zipline l_z, such that the two sides (a, d) and (b, c) of f' that belong to P are perpendicular to l_z and l_b, and the other two sides (a, b) and (d, c) lie in l_z and l_b, respectively. Let C be a cycle of grid edges in tr such that (a, b) belongs to C and is the only edge of C not in P. Then the edges (a, d) and (b, c) of f' together with the edges of C except for (a, b) determine a subpath of P called a *local cookie* with base (d, c) on l_b. Depending on its shape, a local cookie has one of four types: I, T, q_1-facing and q_2-facing. For example, C and f_2' in Fig. 4 make a type T local cookie.

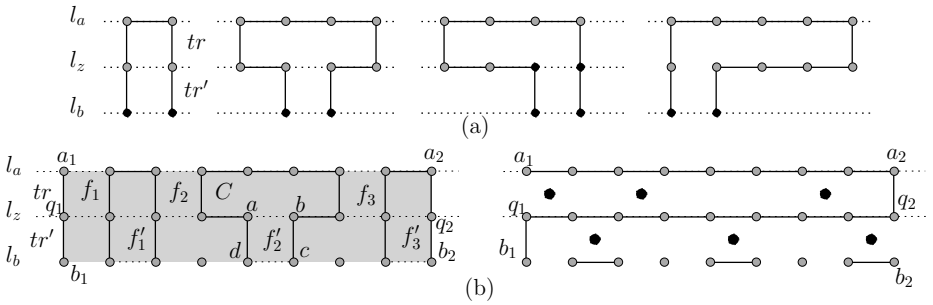

Fig. 4. (a) The types of local cookies. (b) Zip Z turns l_a and l_z into segments. *Left*: R_Z (in grey) before zip Z. Dotted edges do not belong to P. Track tr is covered locally. Cells in S_{tr} and $S_{tr'}$ are labeled f_i and f_i'. *Right*: The big dots mark the switched cells after the zip. On l_b, only edges of P in the f_i's are shown.

The goal of a zip operation is to create a new s,t Hamiltonian path P' that contains all of l_a and l_z as two segments in P' and joins them with a boundary edge of tr in P'. These two segments and the boundary edge that connects them can be viewed as a round trip from one boundary to the opposite boundary and back. This goal motivates the following definition.

Definition 2. *The main track tr is* locally covered *by P provided (i) P contains edge (a_1, q_1) or else contains a regular cookie whose base is the edge (a_1, q_1)– either way, P contains the edge (q_1, b_1), and (ii) P covers the remaining nodes of tr with local cookies with base in l_b (e.g., Fig. 4 (Left)).*

Note that here, either a_1 has degree 1 on path P and lies at a corner of $R_{\mathbb{G}}$, or else a_1 is incident to a boundary edge that belongs to P but lies outside tr.

Definition 3. *The zone R_Z of a zip Z is zippable provided the main track tr is locally covered by P (See Fig. 4 (Left)).*

Observation 6 (Special switchable cells). (a) By Definition 1, each grid cell f' of tr' that contains the base of a local cookie of R_Z is switchable. Switching any such f' creates a cycle-path cover $\mathbb{P} = \{P', C\}$ of $\mathbb{G}$ where cycle C lies in tr. (b) Each grid cell f in the main track tr of a zippable zone R_Z that does not lie inside a local cookie is switchable for P.

We now define two special sets of switchable cells S_{tr} in tr and $S_{tr'}$ in tr' for a zippable zone R_Z and tell how we index the cells.

The set S_{tr} consists of the following cells of tr: any cell that has one side in each of two distinct local cookies; any cell that has a_1q_1 as a side where (a_1, q_1) belongs to P, and has its parallel side in a local cookie; and any cell that has as one side the end of a cookie lying in tr with base a_1, q_1, and has for a parallel side an edge in a local cookie. We index the cells of S_{tr} $f_1 \ldots$ in their order of occurrence from the q_1 end to the q_2 end of tr. We define the set $S_{tr'}$ to be all the cells in tr' that have a side on l_b that is the base of a local cookie. We index the cells of $S_{tr'}$ $f'_1 \ldots$ in order of their occurrence in tr'. We note that $|S_{tr}| = |S_{tr'}|$. We index each local cookie according to the cell f'_i it encloses.

Definition 4 (Zip for zippable R_Z). *Let $l_z^{q_1,q_2}$ be a zipline of $\mathbb{G}$ whose zone R_Z is zippable, and let S_{tr} and $S_{tr'}$ be the special sets of switchable cells in the two tracks tr and tr' of R_Z. The* zip *operation $Z = zip(\mathbb{G}, P, l_z^{q_1,q_2}, tr)$ applies switch to all the cells of S_{tr} and $S_{tr'}$ in the following order: $f_1, f'_1, f_2, f'_2, \ldots$.*

Note that the zip operation is only defined for a zippable zone R_Z. Proofs of correctness of our algorithms will show that the zips are done in zippable zones.

Observation 7. Let P' be the s,t path resulting from a zip Z on an s,t path P of $\mathbb{G}$, where the zone R_Z is zippable. Then, the following hold: (1) Path P' is Hamiltonian and differs from P only in the cells of S_{tr} and $S_{tr'}$; (2) P' contains segments $seg[a_1, a_2]$ on l_a and $seg[q_2, q_1]$ on l_z and the boundary edge (a_2, q_2) joining their end points a_2 and q_2; (3) The boundary edge (q_1, b_1) is the only edge of P' that joins $seg[q_2, q_1]$ to l_b; and (4) P' can be obtained from P in $O(\max\{m, n\})$ switches.

In the next section, we will use path structure (Section 3) to show that doing a zip leaves the next pair of tracks zippable. When this condition holds, we will be able to apply a sequence of zips advancing the zipline by 2 units each time. We refer to such a sequence of zips as a *sweep*. Sweeps can be done in any of the four directions: up, down, left and right.

5 Reconfiguring 1-Complex Paths to Canonical Forms

We give an algorithm called 1ComplexToCanonical to reconfigure any 1-complex s, t path P to a canonical path. In Section 6, we will use this algorithm to design another algorithm to reconfigure between any two 1-complex paths.

We first define some terminology in order to describe the algorithm. A *sub-rectangle* $\mathbb{G}'$ is an $m \times n'$, $n' \leq n$, subgrid of $\mathbb{G}$ such that the subpath P' of P covering the vertices of $\mathbb{G}'$ is a Hamiltonian path between two corner vertices of $\mathbb{G}'$; we call P' a *sub-rectangular path*. We use *sub-rectangle* and *sub-rectangular path* interchangeably for the rest of the section.

We now give a brief overview of the algorithm. It first breaks $\mathbb{G}$ into sub-rectangles using P and the *Structure Theorem* (Theorem 1), so that each sub-rectangle $\mathbb{G}'$ contains an s', t' Hamiltonian path $P'_{s',t'}$ that can be reconfigured to a canonical path of $\mathbb{G}'$ using at most two sweeps in $\mathbb{G}'$. We then merge all the canonical sub-rectangular paths into an s, t Hamiltonian path of $\mathbb{G}$; and using at most one sweep in $\mathbb{G}$, we reconfigure it to a canonical path of $\mathbb{G}$.

Breaking P into (extended) sub-rectangles. We break P into sub-rectangles $\mathbb{G}_h$, $1 \leq h \leq Q$, by removing the following edges: all straight separators, and the edges on $\mathcal{N}$ and $\mathcal{S}$ preceding and following them; the edge between Columns x and $x+1$ on $\mathcal{N}$ or $\mathcal{S}$, when the internal vertices of Column x are completely covered by a separator of form D or I, respectively, or the internal vertices of Column $x+1$ are completely covered by a separator of form J or G. In the path in Fig. 2, removing the bold black edges will break the path into the $Q = 8$ sub-rectangles in Fig. 5(a). We call $\mathbb{G}_1$ and $\mathbb{G}_Q$ the *terminal* sub-rectangles, and the others the *middle* sub-rectangles.

Let s_h and t_h be the starting and ending points of the sub-rectangular path P_h of $\mathbb{G}_h$. If s_h is on $\mathcal{S}$, we flip $\mathbb{G}_h$ along $\mathcal{S}$ when $h < Q$. In case of $\mathbb{G}_Q$, we rotate it by π about its center. If t_h is on $\mathcal{N}$, we add a column to the east of $\mathbb{G}_h$, $1 \leq h \leq Q$, to create the *extended sub-rectangle* $\mathbb{G}'_h$, then connect t_h to the bottom-right corner t'_h of $\mathbb{G}'_h$ through the new edges to get an s_h, t'_h Hamiltonian path of $\mathbb{G}'_h$. From now on, we use $\mathbb{G}_h$ to denote the final sub-rectangle obtained after the optional flipping and/or extending steps. We now observe some properties of the middle and terminal sub-rectangles, then describe Algorithm ReconfigSubRect, to reconfigure a sub-rectangle to a canonical form.

Middle Sub-rectangles. As shown in Fig. 5, each middle sub-rectangle $\mathbb{G}_h$, $1 < h < Q$, must have a corner $\mathcal{W}$ cookie c of unit size. If an $\mathcal{S}$ cookie follows c, then it must be followed by a separator of form D and then the dummy $\mathcal{E}$ boundary. Otherwise, c is followed by a separator of form F, which is the start of the middle subpath of P_h.

Lemma 2. *Let P' be a middle sub-rectangular path. Then any $\mathcal{S}$ cookie of P' has the same parity as m and any $\mathcal{N}$ cookie has even size.*

Terminal sub-rectangles. We limit our discussion of terminal sub-rectangles to $\mathbb{G}_1$, since $\mathbb{G}_Q$ has similar structure. If η_1 of the 1-complex path P of $\mathbb{G}$ has form B or G, then $\mathbb{G}_1$ contains only the initial subpath of P. Otherwise, $\mathbb{G}_1$ contains η_1 of P as the first cross-separator of P_1, where η_1 must have either form D, A or F. If η_1 has form D, then it must also be the last separator, and there are no corner separators or $\mathcal{N}$ cookies in $\mathbb{G}'_s$ by Observation 3. We now assume that η_1 has either form A or F and thus can have cookies in the middle subpath. We show that the size of those cookies depends on the $\mathcal{W}$ compatibility of $\mathbb{R}_s$ and $\mathbb{R}_\alpha$.

Lemma 3. *Let η_1 of $\mathbb{G}_1$ have form A or F. (a) If $\mathbb{R}_s$ is $\mathcal{W}$ compatible, then any $\mathcal{N}$ cookie in the middle subpath of $\mathbb{G}_1$ must have even size, and the size of all $\mathcal{S}$ cookies have opposite parity as m. (b) Otherwise, any $\mathcal{S}$ cookie in the middle subpath must have even size, and the sizes of the $\mathcal{N}$ cookies will have the opposite parity of m.*

Algorithm ReconfigSubRect. If $\mathbb{G}_h$ is a middle sub-rectangle, we apply a *SweepDown* procedure, first placing the zipline on Row 1, and moving down two rows after each zip until we reach the $\mathcal{S}$ boundary. If m is odd, we get an $\mathcal{E}-\mathcal{W}$ canonical path of $\mathbb{G}_h$ at this point. Otherwise, we will end up with unit size $\mathcal{S}$ cookies after the sweep; therefore, we *SweepLeft* to grow the $\mathcal{S}$ cookies all the way to the $\mathcal{N}$ boundary and obtain an $\mathcal{N}-\mathcal{S}$ canonical path of $\mathbb{G}_h$.

If $\mathbb{G}_1$ contains only the initial subpath of P, then we apply a *SweepLeft*, and then a *SweepDown* if we end up with unit size $\mathcal{W}$ cookies in Column 1 after the first sweep. Otherwise, depending on the $\mathcal{W}$ compatibility of η_1, we either *SweepDown* or *SweepUp*, and then we *SweepLeft* if we have unit size $\mathcal{S}$ or $\mathcal{N}$ cookies, respectively, after the first sweep. We can prove the correctness of this algorithm using the Structure Theorem (Theorem 1), and Lemmas 1–3.

Theorem 2. *Algorithm* ReconfigSubRect *reconfigures a sub-rectangle $\mathbb{G}_h$ to a canonical form in $O(|\mathbb{G}_h|)$ switch operations.*

Merging the canonical sub-rectangles. For each $\mathbb{G}_h$, if the $\mathcal{E}$ boundary is a dummy column, and m is odd, we apply one vertical zip with the zipline on Column $n'-2$ of $\mathbb{G}_h$ such that both Columns $n-2$ and $n-1$ become path segments after the zip (Fig. 5(b)). We remove the dummy edges; flip the sub-rectangle back, if it was flipped before; then add all the straight separators and edges on the $\mathcal{N}$ and $\mathcal{S}$ boundaries that were removed, to get an s,t Hamiltonian path P' of $\mathbb{G}$. If P' is not a canonical path, it must have "comb" shaped subpaths connected by straight separators as shown in Fig. 5(c). We then apply one more *SweepDown* to obtain an $\mathcal{E}-\mathcal{W}$ canonical path of $\mathbb{G}$.

Theorem 3. *Algorithm* 1ComplexToCanonical *reconfigures a 1-complex s,t Hamiltonian path to a canonical Hamiltonian path of $\mathbb{G}$ in $O(|\mathbb{G}|)$ time using $O(|\mathbb{G}|)$ switch operations.*

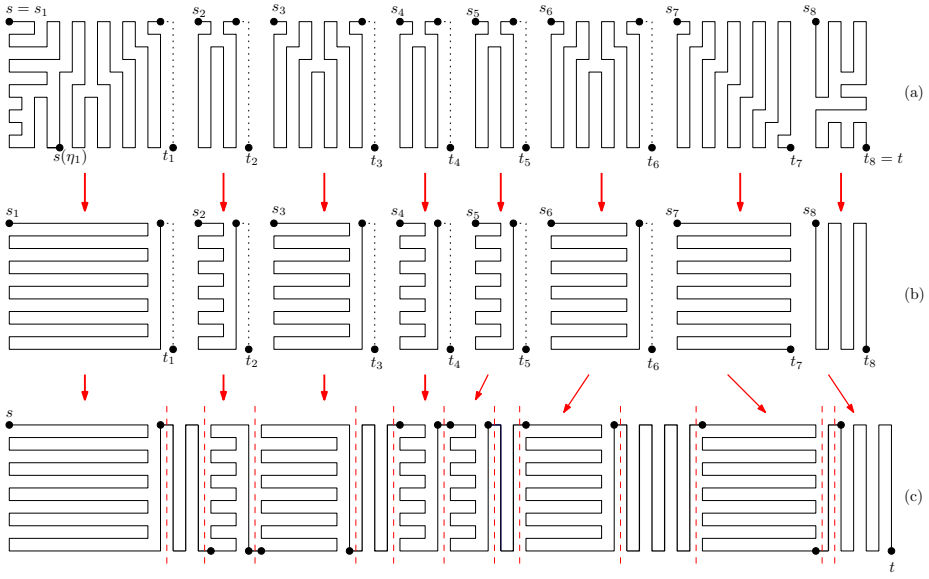

Fig. 5. (a) Extended sub-rectangles of the path in Fig. 2; (b) the canonical forms for the sub-rectangles; (c) the canonical sub-rectangles merged.

6 Reconfiguring between 1-Complex Paths

In this section, we give an algorithm called 1ComplexTo1Complex to reconfigure between any two 1-complex s, t Hamiltonian paths P_1 and P_2 in $O(|\mathbb{G}|)$ time. Our strategy is to use two *canonical Hamiltonian paths* $\mathbb{P}_1$ and $\mathbb{P}_2$ as intermediate paths, where the two canonical paths may or may not be the same based on the parity of m and n. The reconfiguration sequence is as follows: (a) P_1 to $\mathbb{P}_1$, (b) $\mathbb{P}_1$ to $\mathbb{P}_2$ if they are different, and finally (c) $\mathbb{P}_2$ to P_2. Algorithm 1ComplexToCanonical suffices for Steps (a) and (c), since reconfiguring $\mathbb{P}_2$ to P_2 is similar to reversing the steps of the reconfiguration of P_2 to $\mathbb{P}_2$. Now, we give an algorithm that we call CanonicalToCanonical to reconfigure one canonical Hamiltonian path to the other.

Let $\mathbb{P}_1$ and $\mathbb{P}_2$ be the two input canonical paths of $\mathbb{G}$ to Algorithm CanonicalToCanonical. We check the first edge on each path. If the edge is on the $\mathcal{W}$ boundary then the path is an $\mathcal{N}$-$\mathcal{S}$ canonical path; otherwise, it is an $\mathcal{E}$-$\mathcal{W}$ canonical path. If $\mathbb{P}_1$ and $\mathbb{P}_2$ are the same path, then $\mathbb{P}_1$ is returned by the algorithm. Otherwise, if $\mathbb{P}_1$ is $\mathcal{N}$-$\mathcal{S}$ and $\mathbb{P}_2$ is $\mathcal{E}$-$\mathcal{W}$, we apply a *SweepDown* procedure on $\mathbb{P}_1$ starting from Row 1 and ending on Row $m-2$. In the remaining case, when $\mathbb{P}_1$ is $\mathcal{E}$-$\mathcal{W}$ and $\mathbb{P}_2$ is $\mathcal{N}$-$\mathcal{S}$, we apply *SweepLeft* on $\mathbb{P}_1$ with the zipline sweeping from Column 1 to Column $n-2$. To conclude,

Theorem 4. *Let $\mathbb{P}_1$ and $\mathbb{P}_2$ be two canonical paths of $\mathbb{G}$. Then Algorithm* CanonicalToCanonical *reconfigures $\mathbb{P}_1$ to $\mathbb{P}_2$ in $O(|\mathbb{G}|)$ time using $O(|\mathbb{G}|)$ switches.*

Theorem 5. *[**Main algorithmic result.**] Let P_1 and P_2 be two 1-complex s,t Hamiltonian paths of a grid graph $\mathbb{G}$. Then Algorithm* 1ComplexTo1Complex *reconfigures P_1 to P_2 with zips in $O(|\mathbb{G}|)$ time, using $O(|\mathbb{G}|)$ switches.*

7 Conclusion

We established the structure of any 1-complex s,t Hamiltonian path in $\mathbb{G}$. We gave an $O(|\mathbb{G}|)$ algorithm to reconfigure any such path to another using *switches*. It would be interesting to find an algorithm that keeps the s,t Hamiltonian paths in the intermediate steps 1-complex. The reconfiguration problem remains open for grid graphs with arbitrary boundary, and in d-dimension, $d \geq 3$.

References

1. Collins, K.L., Krompart, L.B.: The number of Hamiltonian paths in a rectangular grid. Discrete Mathematics **169**(1–3), 29–38 (1997)
2. Everett, H.: Hamiltonian paths in nonrectangular grid graphs. Master's thesis, University of Saskatchewan, Canada (1986)
3. Fellows, M., Giannopoulos, P., Knauer, C., Paul, C., Rosamond, F.A., Whitesides, S., Yu, N.: Milling a graph with turn costs: A parameterized complexity perspective. In: WG 2010. LNCS, vol. 6410, pp. 123–134. Springer (2010)
4. Gorbenko, A., Popov, V., Sheka, A.: Localization on discrete grid graphs. In: CICA 2011. pp. 971–978. Springer Netherlands, Dordrecht (2012)
5. Itai, A., Papadimitriou, C.H., Szwarcfiter, J.L.: Hamilton paths in grid graphs. SIAM Journal on Computing **11**(4), 676–686 (1982)
6. Jacobsen, J.L.: Exact enumeration of Hamiltonian circuits, walks and chains in two and three dimensions. J. of Phys. A: Math. Theor. **40**, 14667–14678 (2007)
7. Keshavarz-Kohjerdi, F., Bagheri, A.: Hamiltonian paths in L-shaped grid graphs. Theoretical Computer Science **621**, 37–56 (2016)
8. Muller, P., Hascoet, J.Y., Mognol, P.: Toolpaths for additive manufacturing of functionally graded materials (FGM) parts. Rapid Prototyping Journal **20**(6), 511–522 (2014)
9. Nishat, R.I., Srinivasan, V., Whitesides, S.: Reconfiguring simple s,t Hamiltonian paths in rectangular grid graphs. In: IWOCA 2021. LNCS, vol. 12757, pp. 501–515.
10. Nishat, R.I., Whitesides, S.: Reconfiguring Hamiltonian cycles in l-shaped grid graphs. In: WG 2019. LNCS, vol. 11789, pp. 325–337.
11. Nishat, R.I., Whitesides, S.: Bend complexity and Hamiltonian cycles in grid graphs. In: COCOON 2017. LNCS, vol. 10392, pp. 445–456. Springer (2017)
12. Bodroža Pantić, O., Pantić, B., Pantić, I., Bodroža Solarov, M.: Enumeration of Hamiltonian cycles in some grid graphs. MATCH Communications in Mathematical and in Computer Chemistry **70**, 181–204 (2013)
13. Takaoka, A.: Complexity of Hamiltonian cycle reconfiguration. Algorithms **11**(9), 140 (2018)