Sorting on graphs by adjacent swaps using permutation groups

Dohan Kim*

A.I. Research Co., 2537-1 Kyungwon Plaza 201, Sinheung-dong, Sujeong-gu,
Seongnam-si, Kyunggi-do 461-811, South Korea

Abstract

This paper is a review of sorting on several well-known graphs by adjacent swaps using permutation groups. Given a graph with a line, star, complete, or ring topology having n vertices (numbered from 1 to n), we place n objects (numbered from 1 to n) arbitrarily in such a way that exactly one object is placed on each vertex. A sorting procedure is intended to reach the target object placement in which each object i is placed on vertex i for $1 \leq i \leq n$. Now, the problem is to find a minimum-length sequence of adjacent swaps needed to sort the initial arrangement of n objects on n vertices in the graph, which employs a known permutation sorting or permutation decomposition procedure using a generating set of a symmetric group $\mathfrak{S}_n$. This paper reviews the existing permutation sorting and permutation decomposition procedures for sorting on graphs with line, star, complete, and ring topologies by adjacent swaps. We also provide concrete examples and our own implementation for this review paper in order to describe how abstract group-theoretical methods are used to find a minimum-length sequence of adjacent swaps needed to sort objects on those graphs.

Keywords: adjacent swap; object sorting; permutation sorting; permutation group; permutation decomposition.

1 Introduction

Given a graph with a line, star, complete, or ring topology having n vertices, consider that n objects (numbered from 1 to n) are placed on n vertices (numbered from 1 to n) in the graph in such a way that exactly one object is placed on each vertex. This is represented by a bijective assignment between n objects and n vertices, which in turn can be represented by a permutation (i.e. a permutation of n objects). A permutation of n objects can be considered as an element of symmetric group on n letters, denoted by $\mathfrak{S}_n$ [1–3]. Using swaps of objects between two vertices, the goal of a sorting procedure is to sort those objects so that the permutation corresponding to the target assignment is the identity permutation. Sorting plays an important role in algorithm design [4–6], robotics [7–9], graph theory [10–12], genomic rearrangements [13–16], and so on. In particular, sorting objects (e.g. tokens, boxes, markers, tasks, etc.) on graphs has been researched extensively in recent years [7,8,10,17–19]. This review paper is concerned with swaps of objects on adjacent vertices in several well-known graphs, where a swap depends on a graph with a given topology. The basic technique used here is that if a permutation $p = i_1 \cdots i_n \in \mathfrak{S}_n$ denotes an initial bijective assignment between n objects and n vertices, the right multiplication of p by transposition $(s\,t) \in \mathfrak{S}_n$ represents a swap of the object on vertex s and vertex t. Now, the problem of finding a minimum-length sequence of swaps needed to sort n objects is reduced to find a minimum-length permutation decomposition of the inverse of permutation $p \in \mathfrak{S}_n$ using transpositions from the set $T = \{(i\,j) : 1 \leq i < j \leq n\}$. However, if only swaps on adjacent vertices in a graph are allowed, not all transpositions can be used, but transpositions in a chosen generating set of $\mathfrak{S}_n$ can be used. This generating set of $\mathfrak{S}_n$ relies on how vertices are interconnected by a graph with a given topology. In [19] Akers and Krishnamurthy proposed a puzzle consisting of n markers on n vertices in an arbitrary *transposition tree* [20] with n vertices, which is basically the problem of sorting an initial arrangement of n markers on n vertices in a tree with n vertices by adjacent

*E-mail: dkim@airesearch.kr

swaps. They also discussed some tight upper bounds of the minimum number of adjacent swaps needed to sort n markers on n vertices in a certain type of transposition trees [20] with n vertices. In [18] Kim discussed the problem of finding a minimum-length sequence of adjacent task swaps needed from a (bijective) source task assignment to reach a (bijective) target task assignment on a task swapping graph, which is reduced to find a minimum-length sequence of adjacent swaps needed to sort tasks on a task swapping graph. In [10] Yamanaka et al. showed that sorting an initial arrangement n tokens on n vertices in a simple and connected graph of order n by adjacent swaps is solvable in $O(n^2)$ token swaps. They also showed a polynomial-time 2-approximation algorithm for the problem of finding the minimum number of adjacent swaps needed to sort n tokens on n vertices in a tree with n vertices.

The problem of sorting n objects on n vertices by using the minimum number of adjacent swaps in a graph of order n with a line, star, complete, or a ring topology can also be viewed as the problem of finding a *minimum-length generator sequence* [21] of $p^{-1} \in \mathfrak{S}_n$ with respect to the corresponding generating set of $\mathfrak{S}_n$, where a minimum-length generator sequence of a permutation is well-studied in permutation group theory. The minimum-length generator sequence problem is NP-hard in general, but the problem is polynomial for some generating sets and their generated permutations groups.

In this review paper we focus on polynomial-time exact algorithms for sorting by swaps of objects on adjacent vertices in graphs with line, star, complete, and ring topologies. In the remainder of this paper by "sorting objects on a graph" we mean sorting an arrangement of n objects on n vertices in a simple and connected graph of order n. We also assume that both n objects and n vertices in a graph are numbered from 1 to n in such a way that each object is assigned to exactly one vertex, allowing an arrangement of n objects on n vertices to be represented by a permutation, i.e., a permutation $p \in \mathfrak{S}_n$ of n objects.

The remainder of this review paper is organized as follows. Section 2 gives a brief overview of the necessary background used in this review paper. Sorting objects on graphs with line, star, complete, and ring topologies by means of a minimum-length sequence of adjacent swaps is discussed in Section 3. The object sorting procedures described in this section are direct consequences of the known permutation sorting and permutation decomposition (or permutation factorization) methods used in permutation group theory, in which we simply apply them to find a minimum-length sequence of adjacent swaps needed to sort objects on graphs with some well-known topologies. We also discuss upper bounds of the minimum number of adjacent swaps needed to sort objects on those graphs in this section. Section 4 contains a discussion of Cayley graphs for describing the possible arrangements of objects at the vertices of graphs with line, star, complete, and ring topologies. We also discuss the word metric on a Cayley graph in this section, in order to describe the distance between two arrangements of objects at the vertices of the associated graph. Section 5 illustrates some applications of the object sorting procedures discussed in Section 3 to genomic rearrangements. In particular, we discuss sorting permutations (or circular permutations) by reversals of length 2, sorting by prefix-exchanges, and sorting by block-interchanges of length 1 in genomic rearrangements as applications of the object sorting procedures described in Section 3. Discussion and a survey of related works are given in Section 6. In Section 7 we give a brief description of our implementation. We provide concluding remarks in Section 8.

2 Preliminaries

This section describes the necessary background used in this review paper. The definitions and results used in this section are found in [1, 3–5, 10, 18–20, 22–28].

A *group* $(G, \cdot)$ is a nonempty set G, closed under a binary operation $\cdot : G \times G \to G$, such that the following axioms are satisfied: (i) $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ for all $a, b, c \in G$; (ii) there is an element $e \in G$, called an *identity* of G, such that for all $x \in G$, $e \cdot x = x \cdot e = x$; (iii) for each element $a \in G$, there is an element $a^{-1} \in G$ such that $a \cdot a^{-1} = a^{-1} \cdot a = e$. If H is a nonempty subset of G and is also a group under the binary operation $\cdot$ in G, then H is called a *subgroup* of G.

Let $I_n = \{1, 2, \ldots, n\}$. The group of all permutations $I_n \to I_n$ is called the *symmetric group on n letters* and denoted $\mathfrak{S}_n$, in which the binary operation of $\mathfrak{S}_n$ is function composition. A *permutation group* is a subgroup of some $\mathfrak{S}_n$.

Let $i_1, i_2, \ldots, i_n$ be distinct elements of $I_n = \{1, 2, \ldots, n\}$. Then, a permutation that maps $1 \mapsto i_1, 2 \mapsto i_2, \ldots, n \mapsto i_n$ can be written as the *two-line notation* $\begin{pmatrix} 1 & 2 & \cdots & n \\ i_1 & i_2 & \cdots & i_n \end{pmatrix} \in \mathfrak{S}_n$ or the *one-line notation*

$i_1\, i_2\, \cdots\, i_n \in \mathfrak{S}_n$.

Let $i_1, i_2, \ldots, i_r\,(r \leq n)$ be distinct elements of $I_n = \{1, 2, \ldots, n\}$. Then, $(i_1\, i_2 \cdots i_r)$ is defined as the permutation that maps $i_1 \mapsto i_2$, $i_2 \mapsto i_3, \ldots, i_{r-1} \mapsto i_r$ and $i_r \mapsto i_1$, and every other element of I_n maps onto itself. $(i_1\, i_2 \cdots i_r)$ is called a *cycle of length* r or an r*-cycle*. A 2-cycle is called a *transposition*. A 1-cycle is called a *fixed point*.

Every permutation in $\mathfrak{S}_n$ can be written as a product of disjoint cycles. It can also be written as a product of (not necessarily disjoint) transpositions. For example, $p = 5\,4\,3\,6\,1\,2 \in \mathfrak{S}_6$ is written $p = (1\,5)(3)(2\,4\,6)$ as a product of disjoint cycles and $p = (1\,5)(2\,6)(2\,4)$ as a product of transpositions. Unless otherwise stated, we omit fixed points in a product of disjoint cycles, i.e., $p = (1\,5)(2\,4\,6)$.

Let G be a group and let $s_i \in G$ for $i \in I$. The subgroup generated by $S = \{s_i : i \in I\}$ is the smallest subgroup of G containing the set S. If this subgroup is all of G, then S is called a *generating set* of G.

A (*right*) *action* of a group G on a set X is a function $X \times G \to X$ (usually denoted by $(x, g) \mapsto xg$) such that for all $x \in X$ and $g_1, g_2 \in G$: (i) $xe = x$, where e is the identity element of G; (ii) $x(g_1 g_2) = (xg_1)g_2$. When such an action is given, we say that G *acts* (*right*) *on the set* X. If X is G as a set, we say that G acts (right) on itself.

A *minimum-length generator sequence* for permutation p of a permutation group G using the generating set S is a minimum-length sequence consisting of generators in S whose composition is p.

Let S be a generating set of $\mathfrak{S}_n$. A *permutation decomposition* of $p \in \mathfrak{S}_n$ of length l using the generating set S is a product $p = s_1 \cdots s_l$ where each $s_i \in S$ $(1 \leq i \leq l)$.

Let G be a finite group and S be a generating set of G. Then, $\mathrm{Cay}(G, S)$ denotes the *Cayley graph* of G with the generating set S, where the set of vertices V of $\mathrm{Cay}(G, S)$ corresponds to the elements of G, and the set of edges E of $\mathrm{Cay}(G, S)$ corresponds to the action of generators, i.e., $E = \{< x, y >_g : x, y \in G$ and $g \in S$ such that $y = xg\}$.

Recall that the *diameter* of a connected graph is the length of the "longest shortest path" between two vertices of the graph. Let I denote the identity element of G. The diameter of $\mathrm{Cay}(G, S)$ is an upper bound of distance $d(\sigma, I)$ from an arbitrary vertex σ to vertex I in $\mathrm{Cay}(G, S)$, which is an upper bound of the lengths of minimum-length generator sequences for permutations in G using the generating set S.

Consider the following known generating sets of $\mathfrak{S}_n$: $T_1 = \{(i\, i{+}1) : 1 \leq i < n\}, T_2 = \{(1\, i) : 2 \leq i \leq n\}, T_3 = \{(i\, j) : 1 \leq i < j \leq n\}$, and $T_4 = T_1 \cup \{(1\, n)\}$.

The Cayley graph generated by T_1 is called the *bubble-sort graph* BS_n, by T_2 is called the *star graph* ST_n, by T_3 is called the *complete transposition graph* CT_n, and by T_4 is called the *modified bubble-sort graph* MBS_n. The diameters of BS_n, ST_n, CT_n, and MBS_n are $n(n-1)/2$, $\lfloor 3(n-1)/2 \rfloor$, $n-1$, and $\lfloor n^2/4 \rfloor$, respectively [20, 29].

The *inversion number* of p is defined as $|\{(i, j) : i < j, p(i) > p(j)\}|$. The maximum inversion number of permutations of n elements is $n(n-1)/2$, which corresponds to permutation $n\;\, n-1 \cdots 2\, 1$.

A *displacement vector* $d = (d_1, \ldots, d_n)$ of a circular permutation $p \in \mathfrak{S}_n\,(n \geq 3)$ is described as follows. We assume that positions of a circular permutation $p \in \mathfrak{S}_n$ are labelled clockwise starting from 1 (corresponding to the position of the first element of p) to n (corresponding to the position of the last element of p). Note that position 1 and position n are adjacent in a circular permutation $p \in \mathfrak{S}_n$. Each component d_i in d for a circular permutation p is defined as $d_i = j - i$, where $p(j) = i$ for $1 \leq i, j \leq n$. Any displacement vector $d = (d_1, \ldots, d_n)$ satisfies $\sum_{i=1}^{n} d_i = 0$. In particular, each $d_i = 0$ if d is a displacement vector of the identity permutation of $\mathfrak{S}_n$. For example, a displacement vector of a circular permutation $p = 6\,3\,2\,4\,5\,8\,1\,7 \in \mathfrak{S}_8$ is $(6, 1, -1, 0, 0, -5, 1, -2)$. Let d_u be the maximum-valued component of a displacement vector d of a circular permutation $p \in \mathfrak{S}_n$, and d_v be the minimum-valued component of the displacement vector d. In case $d_u - d_v > n$, both d_u and d_v are renewed as $d_u - n$ and $d_v + n$, respectively. This transformation is called the *strictly contracting transformation*. If a displacement vector d admits no strictly contracting transformation, a displacement vector d is called *stable*, denoted by $\bar{d}$. For example, the maximum-valued and minimum-valued component of displacement vector $d = (6, 1, -1, 0, 0, -5, 1, -2)$ of circular permutation $p = 6\,3\,2\,4\,5\,8\,1\,7 \in \mathfrak{S}_8$ are $d_1 = 6$ and $d_6 = -5$, respectively. Since $d_1 - d_6 = 11 > 8$, both d_1 and d_6 are renewed as $d_1 = 6 - 8 = -2$, and $d_6 = -5 + 8 = 3$. This process continues until the stable displacement vector of d is found, i.e., no pair of the maximum-valued component d_u and the minimum-valued component d_v of d admits $d_u - d_v > 8$. Now, we see that $\bar{d} = (-2, 1, -1, 0, 0, 3, 1, -2)$. Based on the stable displacement vector $\bar{d}$, the inversion number $\bar{I}(\bar{d})$ is defined as $\bar{I}(\bar{d}) = |\{(i, j) : (i + \bar{d}_i > j + \bar{d}_j) \vee (i + \bar{d}_i + n < j + \bar{d}_j), 1 \leq i < j \leq n\}|$, which is the minimum length of a permutation decomposition of $p \in \mathfrak{S}_n$ using the generating set $T = \{(i\,i{+}1) : 1 \leq i < n\} \cup \{(1\,n)\}$. We say that the above inversion number for a circular permutation as the *c-inversion number*. Let $path(i)$ denote the path from position i to position k on which element i is located in a circular permutation $p \in \mathfrak{S}_n$, which is uniquely determined by its stable displacement $\bar{d}$. Each $|\bar{d}_i|$ in $\bar{d}$ is interpreted as the length of $path(i)$, where $\bar{d}_i$ is signed positive to denote that $path(i)$ is directed clockwise and signed negative to denote that $path(i)$ is directed counterclockwise.

Theorem 2.1 ([21,26]). *(a) Let $p \in \mathfrak{S}_n$ be a permutation. The minimum number of adjacent swaps needed to sort p is its inversion number.*
(b) Let $q \in \mathfrak{S}_n$ be a circular permutation. The minimum number of adjacent swaps needed to sort q is its c-inversion number.

When considering a swap of objects on vertices in a simple and connected graph of order n (i.e. a graph having n vertices), a swap with *swap distance* $k\,(1 \leq k \leq n)$ is referred to a swap of objects on vertices whose distance is k in a graph. A swap with swap distance 1 is called an *adjacent swap*.

A graph with a line topology is a simple graph in which each vertex is connected to exactly two neighboring vertices other than the two end vertices that are connected to only one neighboring vertex.

A graph with a star topology is a simple graph consisting of a centralized vertex and other vertices in which the centralized vertex connects directly to other vertices and any two vertices except the centralized vertex have no direct connection between each other.

A graph with a complete topology is a simple graph in which every pair of vertices is adjacent.

A graph with a ring topology is a simple cycle graph in which each vertex has direct connections with exactly two other vertices in the topology.

Throughout this review paper we assume that each graph is undirected, simple, and connected having at least three vertices.

3 Adjacent swap-based sorting methods for graphs with line, star, complete, and ring topologies

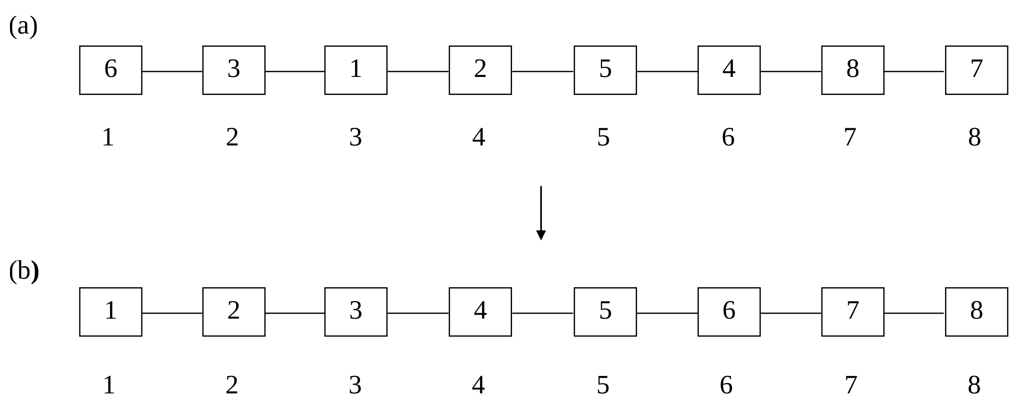

Fig. 1. Sorting eight objects on a graph with a line topology.

We first consider object sorting on a graph with a line topology using adjacent swaps (see Fig. 1). In this review paper we assume that the label of each vertex in a graph denotes an object with that label, while the label next to each vertex in a graph denotes a vertex label. We also assume that each graph with a line topology is labelled in ascending order from left to right (see Fig. 1(a)). Therefore, we see that Fig. 1(a) illustrates the bijective assignment between eight objects (numbered from 1 to 8) and eight vertices (numbered from 1 to 8) in a graph of order eight with a line topology.

Now, the problem is to find a minimum-length sequence of adjacent swaps (i.e. swaps with swap distance 1) needed to sort eight objects in Fig. 1(a). First, we see that permutation $p = 6\,3\,1\,2\,5\,4\,8\,7 \in \mathfrak{S}_8$ represents the initial bijective assignment in Fig. 1(a), while $I = 1\,2\,3\,4\,5\,6\,7\,8 \in \mathfrak{S}_8$ represents the sorted bijective assignment (see Fig. 1(b)). The problem is now reduced to find a minimum-length sequence of permutation sorting by means of transpositions from the set $T' = \{(i\,i{+}1) : 1 \le i < 8\}$. The way how to find a minimum-length sequence of permutation sorting by means of transpositions from the set $T' = \{(i\,i{+}1) : 1 \le i < 8\}$ is to apply the *bubble sort algorithm* [5]. Recall that the bubble sort algorithm sorts permutation $p \in \mathfrak{S}_8$ of eight objects in the following manner:

$$6\,3\,1\,2\,5\,4\,8\,7 \xrightarrow{<1,2>} 3\,6\,1\,2\,5\,4\,8\,7 \xrightarrow{<2,3>} 3\,1\,6\,2\,5\,4\,8\,7 \xrightarrow{<1,2>} 1\,3\,6\,2\,5\,4\,8\,7 \xrightarrow{<3,4>} 1\,3\,2\,6\,5\,4\,8\,7 \xrightarrow{<2,3>} 1\,2\,3\,6\,5\,4\,8\,7 \xrightarrow{<4,5>} 1\,2\,3\,5\,6\,4\,8\,7 \xrightarrow{<5,6>} 1\,2\,3\,5\,4\,6\,8\,7 \xrightarrow{<4,5>} 1\,2\,3\,4\,5\,6\,8\,7 \xrightarrow{<7,8>} 1\,2\,3\,4\,5\,6\,7\,8,$$

where the label $< a, b >$ of each arrow indicates that vertices with vertex labels a and b are involved in each adjacent swap. We see that the above bubble sort algorithm decreases the inversion number by 1 whenever each adjacent swap takes place.

Proposition 3.1. *Given an initial bijective assignment between n objects and n vertices in a graph of order n with a line topology, let $\tau \in \mathfrak{S}_n$ represent the initial bijective assignment between n objects and n vertices.*
(a) The minimum number of adjacent swaps needed to sort n objects is the inversion number of permutation τ.
(b) An upper bound of the minimum number of adjacent swaps needed to sort n objects is $n(n-1)/2$.

Proof. (a) The problem of finding the minimum number of adjacent swaps needed to sort the initial arrangement of n objects on n vertices in the given graph with a line topology is reduced to the problem of finding the minimum number of adjacent swaps needed to sort permutation τ. Since the minimum number of adjacent swaps needed to sort permutation $\tau \in \mathfrak{S}_n$ is its inversion number by Theorem 2.1(a), the result follows.
(b) Since the diameter of the Cayley graph BS_n generated by $T = \{(i\,i{+}1) : 1 \le i < n\}$ is an upper bound of distance $d(\sigma, I)$ from an arbitrary vertex $\sigma \in \mathfrak{S}_n$ to vertex I corresponding to the identity permutation, the result follows from the proof of (a) and the diameter of BS_n, which is $n(n-1)/2$. ☐

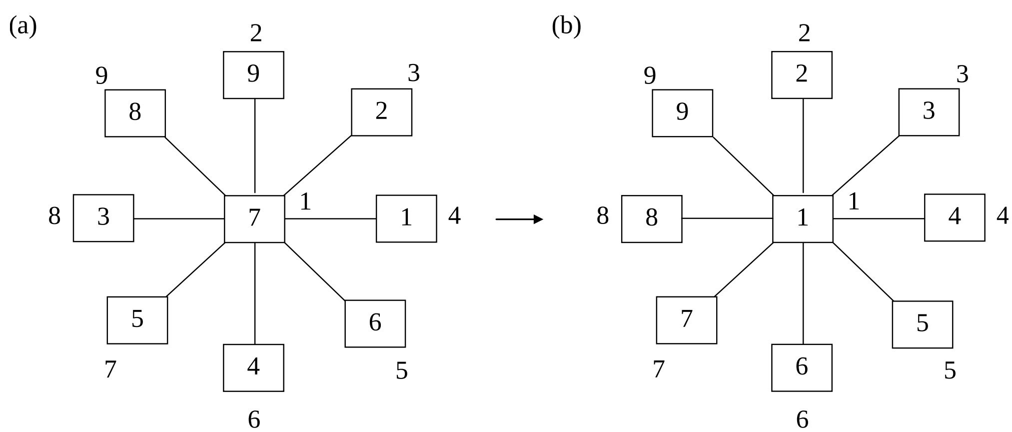

Fig. 2. Sorting nine objects on a graph with a star topology.

We next consider object sorting on a graph with a star topology using adjacent swaps (see Fig. 2). We assume that a centralized vertex in a graph with a star topology is labelled 1, while other vertices are labelled from 2 to n $(n \ge 3)$ in a (clockwise) ascending order. We see that permutation $p = 7\,9\,2\,1\,6\,4\,5\,3\,8 \in \mathfrak{S}_9$ represents the initial bijective assignment in Fig. 2(a), while $I = 1\,2\,3\,4\,5\,6\,7\,8\,9 \in \mathfrak{S}_9$ represents the sorted bijective assignment (see Fig. 2(b)). Since p^{-1} takes p to the identity permutation I (i.e. $pp^{-1} = I$), the problem of finding a minimum-length sequence of adjacent swaps needed to sort nine objects in Fig. 2(a) is reduced to find a minimum-length permutation decomposition of p^{-1} using the set of transpositions $T' = \{(1\,i) : 2 \le i \le 9\}$. The following procedure is to find a minimum-length permutation decomposition of $\pi \in \mathfrak{S}_n$ $(n \ge 3)$ using the set of transpositions $T = \{(1\,i) : 2 \le i \le n\}$.

Procedure 3.2. *PERMUTATION DECOMPOSITION FOR STAR TOPOLOGY [18, 28, 30]*
Input: A permutation $\pi \in \mathfrak{S}_n$ $(n \ge 3)$.

Output: A minimum-length permutation decomposition of π *using the set of transpositions* $T = \{(1\,i) : 2 \leq i \leq n\}$.

- *Write* $\pi \in \mathfrak{S}_n$ *as a product of disjoint cycles in such a manner that* $(1\,p_2 \cdots p_j)(q_1^1 \cdots q_{l_1}^1) \cdots (q_1^m \cdots q_{l_m}^m) \in \mathfrak{S}_n$ *if* $j \geq 2$*, write* π *as* $(q_1^1 \cdots q_{l_1}^1) \cdots (q_1^m \cdots q_{l_m}^m) \in \mathfrak{S}_n$ *otherwise.*
- ***if*** $j \geq 3$*,* ***then*** $(1\,p_2 \cdots p_j)$ *in* π *is decomposed into* $(1\,p_j)(1\,p_{j-1}) \cdots (1\,p_2)$.
- ***for*** $k \leftarrow 1$ ***to*** m ***do***
  - ***if*** $(l_k \geq 2)$ ***then*** $(q_1^k \cdots q_{l_k}^k)$ *in* π *is decomposed into* $(1\,q_1^k)(1\,q_{l_k}^k)(1\,q_{l_k-1}^k) \cdots (1\,q_1^k)$

By applying Procedure 3.2, we find a minimum-length sequence of adjacent swaps needed to sort the initial arrangement of nine objects in Fig. 2(a) represented by permutation $p = 7\,9\,2\,1\,6\,4\,5\,3\,8 \in \mathfrak{S}_9$. First, we see that permutation $p^{-1} = 4\,3\,8\,6\,7\,5\,1\,9\,2 \in \mathfrak{S}_9$ can be written as a product of disjoint cycles, i.e., $p^{-1} = (1\,4\,6\,5\,7)(2\,3\,8\,9)$. Therefore, it is decomposed into $p^{-1} = (1\,7)(1\,5)(1\,6)(1\,4)(1\,2)(1\,9)(1\,8)(1\,3)(1\,2)$ using the set of transpositions $T' = \{(1\,i) : 2 \leq i \leq 9\}$ by Procedure 3.2. Now, the initial bijective assignment in Fig. 2(a) represented by permutation $p = 7\,9\,2\,1\,6\,4\,5\,3\,8 \in \mathfrak{S}_9$ of nine objects is sorted as follows by means of the minimum number of adjacent swaps:

$$7\,9\,2\,1\,6\,4\,5\,3\,8 \xrightarrow{<1,7>} 5\,9\,2\,1\,6\,4\,7\,3\,8 \xrightarrow{<1,5>} 6\,9\,2\,1\,5\,4\,7\,3\,8 \xrightarrow{<1,6>} 4\,9\,2\,1\,5\,6\,7\,3\,8 \xrightarrow{<1,4>} 1\,9\,2\,4\,5\,6\,7\,3\,8 \xrightarrow{<1,2>}$$
$$9\,1\,2\,4\,5\,6\,7\,3\,8 \xrightarrow{<1,9>} 8\,1\,2\,4\,5\,6\,7\,3\,9 \xrightarrow{<1,8>} 3\,1\,2\,4\,5\,6\,7\,8\,9 \xrightarrow{<1,3>} 2\,1\,3\,4\,5\,6\,7\,8\,9 \xrightarrow{<1,2>} 1\,2\,3\,4\,5\,6\,7\,8\,9,$$

where the label $< a, b >$ of each arrow indicates that vertices with vertex labels a and b are involved in each adjacent swap.

Proposition 3.3. *Given an initial bijective assignment between* n *objects and* n *vertices in a graph of order* n *with a star topology, let* $\tau \in \mathfrak{S}_n$ *be a permutation representing the initial bijective assignment and also let* τ^{-1} *be written in a ususal way as a product of disjoint cycles: (i)* $(1\,p_2 \cdots p_j)(q_1^1 \cdots q_{l_1}^1) \cdots (q_1^m \cdots q_{l_m}^m) \in \mathfrak{S}_n$ *if* $j \geq 2$*; (ii)* $(q_1^1 \cdots q_{l_1}^1) \cdots (q_1^m \cdots q_{l_m}^m) \in \mathfrak{S}_n$ *otherwise.*
(a) The minimum number of adjacent swaps needed to sort n *objects is* $n+m-r-1$ *for case (i) and* $n+m-r$ *for case (ii), where* r *is the number of fixed points in* τ^{-1}.
(b) An upper bound of the minimum number of adjacent swaps needed to sort n *objects is* $\lfloor 3(n-1)/2 \rfloor$.

Proof. (a) We reduce the problem of finding the minimum number of adjacent swaps needed to sort the initial arrangement of n objects on n vertices in the given graph with a star topology to the problem of finding the minimum length of a permutation decomposition of τ^{-1} using the set of transpositions $T = \{(1\,i) : 2 \leq i \leq n\}$. Since the minimum length of a permutation decomposition of a permutation written as a product of disjoint cycles as above using the set of transpositions $T = \{(1\,i) : 2 \leq i \leq n\}$ is $n+m-r-1$ for case (i) and $n+m-r$ for case (ii) [28], the result follows.
(b) Since the diameter of the Cayley graph ST_n generated by $T = \{(1\,i) : 2 \leq i \leq n\}$ is an upper bound of distance $d(\sigma, I)$ from an arbitrary vertex $\sigma \in \mathfrak{S}_n$ to vertex I corresponding to the identity permutation, the result follows from the proof of (a) and the diameter of ST_n, which is $\lfloor 3(n-1)/2 \rfloor$. □

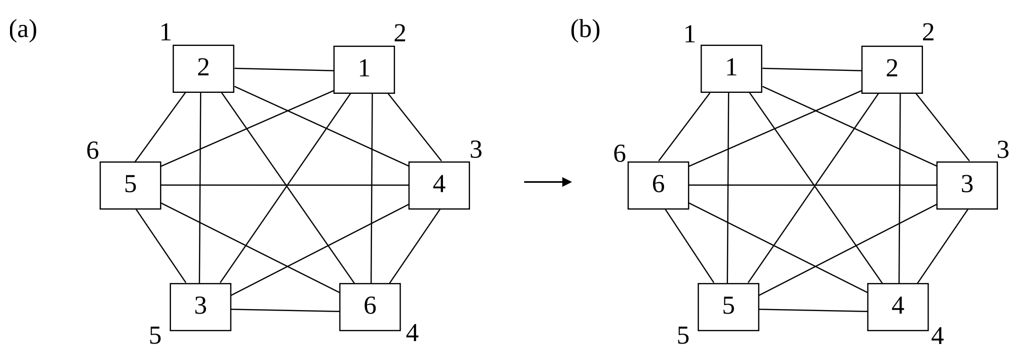

Fig. 3. Sorting six objects on a graph with a complete topology.

We next consider object sorting on a graph with a complete topology using adjacent swaps (see Fig. 3). We assume that each vertex in a graph with a complete topology is labelled from 1 to $n\,(n \geq 3)$ in an arbitrary manner. We see that permutation $p = 2\,1\,4\,6\,3\,5 \in \mathfrak{S}_6$ represents the initial bijective assignment in Fig. 3(a), while $I = 1\,2\,3\,4\,5\,6 \in \mathfrak{S}_6$ represents the sorted bijective assignment (see Fig. 3(b)). Now, the problem of finding a minimum-length sequence of adjacent swaps needed to sort six objects in Fig. 3(a) is reduced to find a minimum-length permutation decomposition of p^{-1} using the set of transpositions $T' = \{(i\,j) : 1 \leq i < j \leq 6\}$. The following procedure is to find a minimum-length permutation decomposition of permutation $\pi \in \mathfrak{S}_n\,(n \geq 3)$ using the set of transpositions $T = \{(i\,j) : 1 \leq i < j \leq n\}$.

Procedure 3.4. PERMUTATION DECOMPOSITION FOR COMPLETE TOPOLOGY [18,31,32]
Input: A permutation $\pi \in \mathfrak{S}_n\,(n \geq 3)$.
Output: A minimum-length permutation decomposition of π using the set of transpositions $T = \{(i\,j) : 1 \leq i < j \leq n\}$.

- Write $\pi \in \mathfrak{S}_n$ as a product of disjoint cycles $(q_1^1 \cdots q_{l_1}^1) \cdots (q_1^m \cdots q_{l_m}^m)$.
- **for** $k \leftarrow 1$ **to** m **do**
  - **if** $(l_k \geq 3)$ **then** $(q_1^k \cdots q_{l_k}^k)$ in π is decomposed into $(q_1^k\, q_{l_k}^k)(q_1^k\, q_{l_{k-1}}^k) \cdots (q_1^k\, q_2^k)$

Using Procedure 3.4, we find a minimum-length sequence of adjacent swaps needed to sort the initial arrangement of six objects in Fig. 3(a) represented by permutation $p = 2\,1\,4\,6\,3\,5 \in \mathfrak{S}_6$. We see that permutation $p^{-1} = 2\,1\,5\,3\,6\,4 \in \mathfrak{S}_6$ can be written as a product of disjoint cycles, i.e., $p^{-1} = (1\,2)(3\,5\,6\,4)$. Therefore, it is decomposed into $p^{-1} = (1\,2)(3\,4)(3\,6)(3\,5)$ using the set of transpositions $T' = \{(i\,j) : 1 \leq i < j \leq 6\}$ by Procedure 3.4. Now, the initial bijective assignment in Fig. 3(a) represented by permutation $p = 2\,1\,4\,6\,3\,5 \in \mathfrak{S}_6$ of six objects is sorted as follows by means of the minimum number of adjacent swaps:

$$2\,1\,4\,6\,3\,5 \xrightarrow{<1,2>} 1\,2\,4\,6\,3\,5 \xrightarrow{<3,4>} 1\,2\,6\,4\,3\,5 \xrightarrow{<3,6>} 1\,2\,5\,4\,3\,6 \xrightarrow{<3,5>} 1\,2\,3\,4\,5\,6,$$

where the label $< a, b >$ of each arrow indicates that vertices with vertex labels a and b are involved in each adjacent swap.

Proposition 3.5. *Given an initial bijective assignment between n objects and n vertices in a graph of order n with a complete topology, let $\tau \in \mathfrak{S}_n$ be a permutation representing the initial bijective assignment between n objects and n vertices and let τ^{-1} be written as a product of disjoint cycles.*
(a) The minimum number of adjacent swaps needed to sort n objects is $n - r$, where r is the number of disjoint cycles in τ^{-1} including fixed points.
(b) An upper bound of the minimum number of adjacent swaps needed to sort n objects is $n - 1$.

Proof. (a) We reduce the problem of finding the minimum number of adjacent swaps needed to sort the initial arrangement of n objects on n vertices in the given graph with a complete topology to the problem of finding the minimum length of a permutation decomposition of τ^{-1} using the set of transpositions $T = \{(i\,j) : 1 \leq i < j \leq n\}$. Since the minimum length of a permutation decomposition of τ^{-1} using the set of transpositions $T = \{(i\,j) : 1 \leq i < j \leq n\}$ is $n - r$ [31], the result follows.
(b) Since the diameter of the Cayley graph CT_n generated by $T = \{(i\,j) : 1 \leq i < j \leq n\}$ is an upper bound of distance $d(\sigma, I)$ from an arbitrary vertex $\sigma \in \mathfrak{S}_n$ to vertex I corresponding to the identity permutation, the result follows from the proof of (a) and the diameter of CT_n, which is $n - 1$. □

We next consider object sorting on a graph with a ring topology by adjacent swaps (see Fig. 4). We assume that each vertex in a graph with a ring topology is labelled in ascending order (clockwise) starting from 1 to $n\,(n \geq 3)$. We also see that permutation $p = 6\,3\,2\,4\,5\,8\,1\,7 \in \mathfrak{S}_8$ represents the initial bijective assignment in Fig. 4(a), while $I = 1\,2\,3\,4\,5\,6\,7\,8 \in \mathfrak{S}_8$ represents the sorted bijective assignment (see Fig. 4(b)). Now, the problem of finding a minimum-length sequence of adjacent swaps needed to sort eight objects in Fig. 4(a) is reduced to find a minimum-length sequence of permutation sorting of $p \in \mathfrak{S}_8$ by means of transpositions from the set $T' = \{(i\,i{+}1) : 1 \leq i < 8\} \cup \{(1\,8)\}$. The following procedure finds a minimum-length sequence of permutation sorting of $\pi \in \mathfrak{S}_n\,(n \geq 3)$ by means of transpositions from the set $T = \{(i\,i{+}1) : 1 \leq i < n\} \cup \{(1\,n)\}$.

Procedure 3.6. PERMUTATION SORTING FOR RING TOPOLOGY [18,21]
Input: A circular permutation $\pi \in \mathfrak{S}_n\,(n \geq 3)$.
Output: A minimum-length sequence of permutation sorting of π by means of transpositions from the set $T = \{(i\,i{+}1) : 1 \leq i < n\} \cup \{(1\,n)\}$.

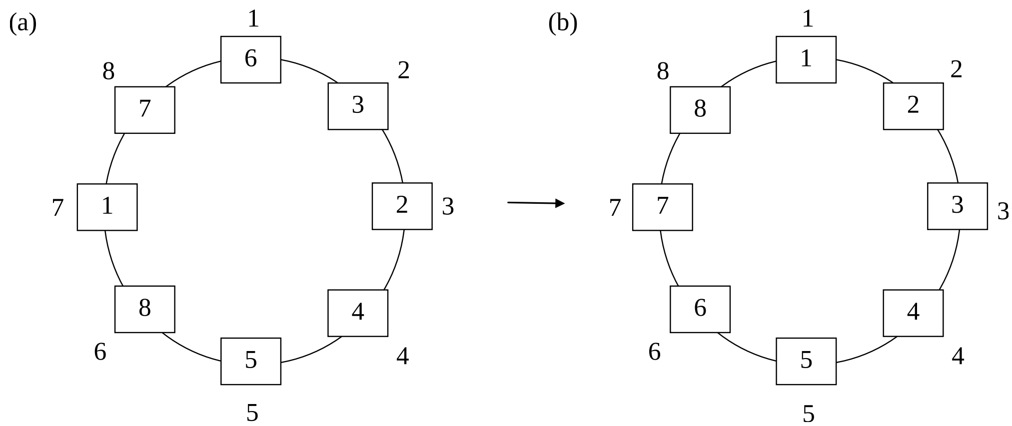

Fig. 4. Sorting eight objects on a graph with a ring topology.

- Compute the stable displacement vector $\bar{d}$ of π. Set $p := \pi$.
- **while** $p \neq I$ **do**
  - Find an adjacent pair of positions a_1 having an element b_1 and a_2 having an element b_2 such that $path(b_1)$ and $path(b_2)$ are directed oppositely from one another having an intersection of both positions a_1 and a_2. If no such an adjacent pair of positions exist, find an adjacent pair of positions a_1 having an element b_1 and a_2 having an element b_2 such that b_1 is homed (i.e. $p(b_1) = b_1$) while b_2 is not homed in which $path(b_2)$ passes through position a_1.
  - Append transposition $(a_1\, a_2)$ to the list for the permutation sorting of π. Swap element b_1 on position a_1 and element b_2 on position a_2 in p. Compute the stable displacement vector of (the updated) p along with (the updated) $path(i)$ for each $1 \leq i \leq n$.

By applying Procedure 3.6, the initial bijective assignment in Fig. 4(a) represented by permutation $p = 6\,3\,2\,4\,5\,8\,1\,7 \in \mathfrak{S}_8$ of eight objects is sorted as follows by means of the minimum number of adjacent swaps:

$$6\,3\,2\,4\,5\,8\,1\,7 \xrightarrow{<2,3>} 6\,2\,3\,4\,5\,8\,1\,7 \xrightarrow{<7,8>} 6\,2\,3\,4\,5\,8\,7\,1 \xrightarrow{<1,8>} 1\,2\,3\,4\,5\,8\,7\,6 \xrightarrow{<7,8>} 1\,2\,3\,4\,5\,8\,6\,7 \xrightarrow{<6,7>}$$
$$1\,2\,3\,4\,5\,6\,8\,7 \xrightarrow{<7,8>} 1\,2\,3\,4\,5\,6\,7\,8,$$

where the label $< a, b >$ of each arrow indicates that vertices with vertex labels a and b are involved in each adjacent swap.

Proposition 3.7. *Let $\tau \in \mathfrak{S}_n$ be a permutation representing an initial bijective assignment between n objects and n vertices in a graph of order n with a ring topology, where $\bar{d}$ is the stable displacement vector of τ and $\bar{I}(\bar{d})$ is its corresponding c-inversion number.*
(a) The minimum number of adjacent swaps needed to sort n objects is $\bar{I}(\bar{d})$.
(b) An upper bound of the minimum number of adjacent swaps needed to sort n objects is $\lfloor n^2/4 \rfloor$.

Proof. (a) The problem of finding the minimum number of adjacent swaps needed to sort the initial arrangement of n objects on n vertices in the given graph with a ring topology is reduced to the problem of finding the minimum number of adjacent swaps needed to sort the circular permutation τ. Since the minimum number of adjacent swaps needed to sort the circular permutation $\tau \in \mathfrak{S}_n$ is its c-inversion number by Theorem 2.1(b), the result follows.
(b) Since the diameter of the Cayley graph MBS_n generated by $T = \{(i\, i{+}1) : 1 \leq i < n\} \cup \{(1\, n)\}$ is an upper bound of distance $d(\sigma, I)$ from an arbitrary vertex $\sigma \in \mathfrak{S}_n$ to vertex I corresponding to the identity permutation, the result follows from the proof of (a) and the diameter of MBS_n, which is $\lfloor n^2/4 \rfloor$. □

4 Cayley graphs as state diagrams for sorting objects on graphs with line, star, complete, and ring topologies

Cayley graphs [3, 19, 20] play an important role in both mathematics (e.g. combinatorics [33, 34], group theory [35, 36], etc.) and computer science (e.g. interconnection network design [3, 19, 20], data center design [37, 38], distributed systems [39, 40], multi-agent systems [18, 41], etc.). In Section 3 we used the diameters of several kinds of Cayley graphs to find upper bounds of the minimum number of adjacent swaps needed to sort objects on graphs with line, star, complete, and ring topologies. This section discusses in detail how different kinds of Cayley graphs serve as state diagrams for sorting objects on those graphs by adjacent swaps. This section also discusses how the word metric is used to describe the distance between two permutations representing two arrangements of objects at the vertices of a graph with a line, star, complete, or ring topology. We first summarize necessary definitions and results used in this section, which are found in [3, 18–20, 36].

Let G be a finite group and S be a generating set of G. Recall that the vertices of a Cayley graph $\mathrm{Cay}(G, S)$ are the elements of G, and the edges are all ordered pairs (g, gs) for $g \in G, s \in S$. The edge (g, gs) is said to be *labelled by* s. If S is closed under inverses, then the edges of $\mathrm{Cay}(G, S)$ are undirected.

A path p from vertex v_1 to vertex v_2 in $\mathrm{Cay}(G, S)$ can be written as a sequence of generators $g_1, \ldots, g_k$, $g_i \in S$, or simply written as $p = g_1 \cdots g_k$.

An *automorphism* of a simple connected graph $\Gamma = (V, E)$ is a permutation p of V such that (u, v) is an edge of Γ if and only if $(p(u), p(v))$ is an edge of Γ.

A simple connected graph $\Gamma = (V, E)$ is said to be *vertex transitive* if given any pair of vertices u and v, there exists an automorphism $\alpha \in \mathrm{Aut}(\Gamma)$ such that $v = \alpha(u)$.

Each element $g \in G$ in $\mathrm{Cay}(G, S)$ can be written as a product of generators $g = g_1 \cdots g_k, g_i \in S$. (A word $g_1 \cdots g_k$ for $k = 0$ is read as the empty word, which is the identity element of G.) If k is minimal among all such expressions for g, then k is called the *length* of g, denoted by $l(g) = k$, and the word $g_1 \cdots g_k$ is called a *reduced word* or *reduced decomposition* for g.

Let X be a set and $d : X \times X \to \mathbb{R}^+$ a function from $X \times X$ to the set $\mathbb{R}^+$ of non-negative real numbers satisfying the following properties. For all x, y, z in X: (a) $d(x, y) = 0$ if and only if $x = y$; (b) $d(x, y) = d(y, x)$; (c) $d(x, z) \leq d(x, y) + d(y, z)$. Then, d is called a *metric* or *distance function* on X and $d(x, y)$ is called the *distance* from x to y. The set X with metric d is called a *metric space* and is denoted by (X, d).

Recall that a metric on $\mathrm{Cay}(G, S)$ can be defined in such a way that the length of every edge in $\mathrm{Cay}(G, S)$ is 1 and the distance between two vertices is the minimum length of edges joining them (i.e. the length of a shortest path between two vertices). The metric on $\mathrm{Cay}(G, S)$ induces a metric on G in a natural way by considering G as the set of vertices in $\mathrm{Cay}(G, S)$. This metric on G is called the *word metric* with respect to S. By the word metric on $\mathrm{Cay}(G, S)$ we mean the word metric on G with respect to S. The word metric on the bubble-sort graph BS_n is called as *Kendall's tau distance* [2], while the word metric on the complete transposition graph CT_n is called as *Cayley distance* [14]. (The interested reader may refer to [2, 42] for other metrics on permutations, such as *Permutation Hamming distance* [42] and *Ulam's distance* [2].)

Since every Cayley graph is vertex transitive [19], we see that a path p from vertex v_1 to vertex v_2 in $\mathrm{Cay}(G, S)$ is also a path from vertex $v_2^{-1}v_1$ to vertex I, where I is the identity element of $\mathfrak{S}_n$. Note that p is also a path from vertex I to vertex $v_1^{-1}v_2$ in $\mathrm{Cay}(G, S)$. To find a shortest path from v_1 to v_2 in $\mathrm{Cay}(G, S)$ is equivalent to find a shortest path from $v_2^{-1}v_1$ to vertex I, which in turn is equivalent to find a reduced word of $v_1^{-1}v_2$. We also see that there are often multiple shortest paths from vertex v_1 to vertex v_2 in $\mathrm{Cay}(G, S)$. For example, since there are 16 reduced words for $4321 \in \mathfrak{S}_4$ with its generating set $S = \{(i\ i{+}1) : 1 \leq i < 4\}$ [36] , there are 16 shortest paths from vertex I to vertex $4321 \in \mathfrak{S}_4$ in BS_4. (See Fig. 5 for the bubble-sort graph BS_4.)

Let $L(n)$ (resp. $R(n)$) denote a graph of order n with a line topology (resp. ring topology) and let $p \in \mathfrak{S}_n$ represent an initial arrangement of n objects on n vertices in $L(n)$ (resp. $R(n)$). The Cayley graph BS_n (resp. MBS_n) can be seen as a state diagram for sorting n objects on $L(n)$ (resp. $R(n)$) by adjacent swaps. The vertices of BS_n (resp. MBS_n) represent the possible arrangements of n objects on n vertices in $L(n)$ (resp. $R(n)$), while the edges of BS_n (resp. MBS_n) correspond to adjacent swaps of objects on $L(n)$ (resp. $R(n)$). Sorting the initial arrangement of n objects on n vertices in $L(n)$ (resp. $R(n)$) by adjacent swaps corresponds to find a path from p to I in BS_n (resp. MBS_n). Furthermore, sorting the initial

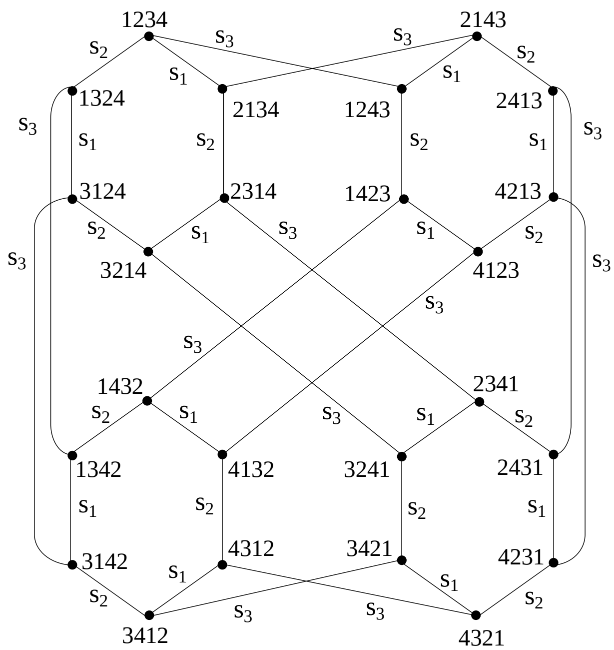

Fig. 5. Bubble-sort graph BS_4 [43].

arrangement of n objects at the vertices of $L(n)$ (resp. $R(n)$) by using the minimum number of adjacent swaps corresponds to find a shortest path from p to I in BS_n (resp. MBS_n), which in turn corresponds to find a minimum-length generator sequence or reduced word of p^{-1} in $\mathfrak{S}_n$ with respect to the generating set $S = \{(i\ i{+}1) : 1 \le i < n\}$ (resp. $S' = S \cup \{(1\,n)\}$).

For example, let $s_i = (i\,i{+}1), 1 \le i < 8$, which generates $\mathfrak{S}_8$. If $q = 6\,3\,1\,2\,5\,4\,8\,7$ (see Fig. 1) represents an initial arrangement of eight objects at the vertices of $L(8)$, a sequence of adjacent swaps represented by $(< 1,2 >, < 2,3 >, < 1,2 >, < 3,4 >, < 2,3 >, < 4,5 >, < 5,6 >, < 4,5 >, < 7,8 >)$ corresponds to a reduced word $s_1 s_2 s_1 s_3 s_2 s_4 s_5 s_4 s_7$ for q^{-1}, which represents a minimum-length sequence of adjacent swaps needed to sort the initial arrangement of eight objects on $L(8)$.

Similarly, let $S(n)$ (resp. $C(n)$) denote a graph of order n with a star topology (resp. complete topology) and let $p \in \mathfrak{S}_n$ be a permutation representing an initial arrangement of n objects on n vertices in $S(n)$ (resp. $C(n)$). We also see that the Cayley graph ST_n (resp. CT_n) can be viewed as a state diagram of sorting n objects on n vertices in $S(n)$ (resp. $C(n)$) by adjacent swaps. The vertices of ST_n (resp. CT_n) represent the possible arrangements of n objects at the vertices of $S(n)$ (resp. $C(n)$), while the edges of ST_n (resp. CT_n) correspond to adjacent swaps of objects on $S(n)$ (resp. $C(n)$). Sorting the initial arrangement of n objects at the vertices of $S(n)$ (resp. $C(n)$) by using the minimum number of adjacent swaps corresponds to find a shortest path from p to I in ST_n (resp. CT_n), which again corresponds to find a minimum-length permutation decomposition or reduced word of p^{-1} in $\mathfrak{S}_n$ with respect to the generating set $T = \{(1\ i) : 2 \le i \le n\}$ (resp. $T' = \{(i\ j) : 1 \le i < j \le n\}$). For example, let $s_i = (1\,i), 2 \le i \le 4$, which generates $\mathfrak{S}_4$. If $q = 2\,3\,4\,1$ represents an initial arrangement of four objects at the vertices of $S(4)$, a sequence of adjacent swaps represented by $(< 1,2 >, < 1,3 >, < 1,4 >)$ sorts q. Since a minimum-length permutation decomposition of $q^{-1} = 4\,1\,2\,3$ is $q^{-1} = (1\,2)(1\,3)(1\,4) = s_1 s_2 s_3$, we see that $s_1 s_2 s_3$ corresponds to a minimum-length sequence of adjacent swaps needed to sort the initial arrangement of four objects on $S(4)$ (see Fig. 6).

Remarks. Algorithms for finding disjoint paths in Cayley graphs were studied in [44–49]. For example, node-disjoint paths between two vertices in the bubble-sort graph BS_n provide intermediate disjoint arrangements of n objects (i.e. distinct permutations of n objects) when transforming one arrangement of n objects into the other arrangement of n objects at the vertices of $L(n)$ by adjacent swaps. Note that those node-disjoint paths are not necessarily shortest paths. In [20] Lakshmivarahan et al. remarked that the prob-

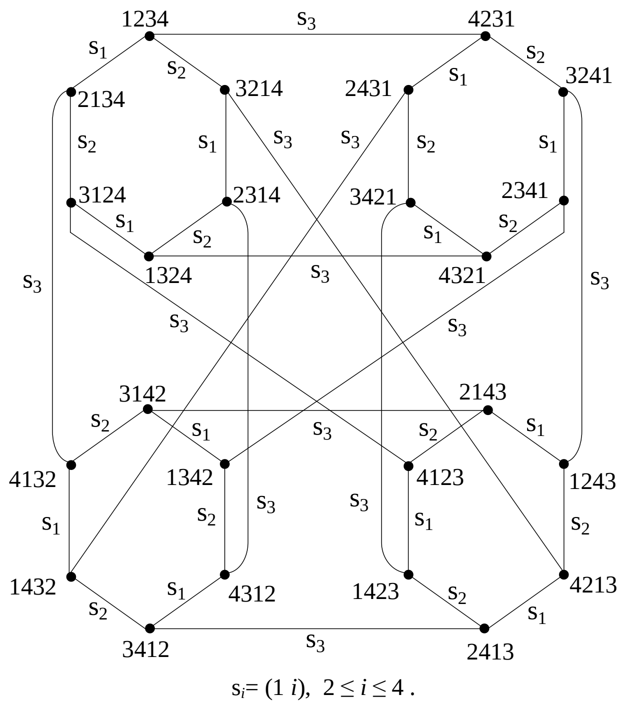

Fig. 6. Star graph ST_4 [3].

lem of enumerating node-disjoint paths in MBS_n is open. The interested reader may refer to [20, 44–47, 50] for further details of disjoint paths in Cayley graphs.

5 Applications to genomic rearrangements

In biology a *genome* is defined as "the entire collection of genes encoded by a particular organism" [51]. A series of genomic rearrangement events may alter the genomic architecture of a species, where each genomic rearrangement event can change the ordering of genes [52]. The main genomic rearrangement operations include *transpositions* [16, 53, 54], *reversals* (also called *inversions*) [55–57], *block-interchanges* [13, 58, 59], etc. Genomes are often represented by permutations of genes. Specifically, when the information about the orientation of genes is not available, unsigned permutations of genes are used for genomic rearrangements [14]. Genomes are represented by unsigned permutations in the situation where (i) the order of genes within each genome is available, (ii) every genome shares the same collection of genes, (iii) every genome has a single copy of each gene, and (iv) every genome consists of a single chromosome [14]. In [15] Christie pointed out that the simplification (iii) is biologically valid when a genome is unlikely to have duplicate genes. We also need to consider circular permutations because bacteria often have a genome consisting of a circular chromosome, which can be naturally represented by a circular permutation [60]. In the remainder of this section we only consider unsigned permutations (resp. unsigned circular permutations) and simply call them permutations (resp. circular permutations).

When genomes are represented by permutations, the *genomic rearrangement problem* [14] is given as "Given two genomes represented by permutations $p_1 \in \mathfrak{S}_n$ and $p_2 \in \mathfrak{S}_n$, find a minimum-length sequence of allowed rearrangement operations that transforms p_1 into p_2 (or p_2 into p_1)." This minimum length is called the *distance* [15] between two genomes. Among the genomic rearrangement operations, we first describe the *transposition*[1] [14, 53] operation of genomic rearrangements.

Let $p = p_1 \cdots p_n \in \mathfrak{S}_n$. A *segment* X is defined as a sequence of consecutive elements $p_i, \ldots, p_j$ $(j \geq i)$. Two segments $X = p_i, \ldots, p_j$ and $Y = p_k, \ldots, p_l$ are contiguous if $k = j + 1$ or $i = l + 1$. A *transposition* on

[1]A transposition used in genomic rearrangements is different from a transposition used in previous sections which is a cycle of order two. The distinction is clear from context.

p is defined as an exchange of two disjoint contiguous segments. Given $p \in \mathfrak{S}_n$, the transposition $T(i,j,k)$ with $1 \le i < j < k \le n+1$ applied to p swaps two disjoint contiguous segments determined respectively by i and $j-1$ and by j and $k-1$, transforming p into $p \cdot T(i,j,k)$. Therefore, $T(i,j,k)$ is described by the following two-line permutation:

$$\begin{pmatrix} 1\cdots i-1 \boxed{i\ i+1\cdots j-2\ j-1}\ \boxed{j\ j+1\cdots k-1}\ k\cdots n \\ 1\cdots i-1 \boxed{j\ j+1\cdots k-1}\ \boxed{i\ i+1\cdots j-2\ j-1}\ k\cdots n \end{pmatrix}.$$

Let $p = p_1 \cdots p_n \in \mathfrak{S}_n$ be a permutation representing a genome. The *reversal* $\rho(i,j)$ of an interval $[i,j]$, $1 \le i < j \le n$, is the permutation [14,55] given as follows:

$$\begin{pmatrix} 1\cdots i-1\ \underline{i\ i+1\cdots j-1\ j}\ j+1\cdots n \\ 1\cdots i-1\ j\ j-1\cdots i+1\ i\ j+1\cdots n \end{pmatrix}.$$

We see that $p \cdot \rho(i,j)$ has the effect of reversing genes $p_i, p_{i+1}, \ldots, p_j$ in a genome represented by p. (A reversal on a circular permutation is defined in a similar way to the above. See [61] for details.)

Meanwhile, block-interchanges [58, 59] can be viewed as generalized transpositions in that they swap two disjoint but not necessarily contiguous segments on a permutation. A block-interchange operation of genomic rearrangements is defined as follows [14, 59]:

Given $p \in \mathfrak{S}_n$, the *block-interchange* $B(i,j,k,l)$ with $1 \le i < j \le k < l \le n+1$ applied to p swaps two disjoint segments determined respectively by i and $j-1$ and by k and $l-1$, transforming p into $p \cdot B(i,j,k,l)$. Therefore, $B(i,j,k,l)$ is described by the following two-line permutation:

$$\begin{pmatrix} 1\cdots i-1 \boxed{i\cdots j-1}\ j\ j+1\cdots k-1 \boxed{k\cdots\ l-1}\ l\ l+1\cdots n \\ 1\cdots i-1 \boxed{k\cdots\ l-1}\ j\ j+1\cdots k-1 \boxed{i\cdots j-1}\ l\ l+1\cdots n \end{pmatrix}.$$

In [58] Lin et al. remarked that block-interchange rearrangements seemingly play an important role in the evolution of bacterial (Vibrio) species. In [59] Chin et al. discussed the *block-interchange distance problem*, which is concerned with finding the minimum number of block-interchanges needed to transform one permutation into another permutation. If the size of blocks to be swapped in block-interchanges is exactly 1, the block-interchanges become the swaps of two elements on a permutation. In particular, if two elements to be swapped are contiguous, those swaps are called *adjacent swaps*[2] [13,62,63]. An adjacent swap can also be viewed as a reversal of length two. It is known that some mutations may cause two adjacent genes in a genome to be reversed [63].

Similarly, the *reversal distance problem* is concerned with finding the minimum number of reversals needed to transform one permutation into another permutation [15].

In Section 3 we discussed sorting n objects on n vertices in a graph of order n with a line or ring topology by adjacent swaps. In this section we consider objects as genes and an arrangement of objects at the vertices of a graph with a line topology (resp. ring topology) as a genome (resp. circular genome). Let $p = 6\,1\,2\,4\,8\,5\,3\,7 \in \mathfrak{S}_8$ be a circular permutation representing a circular genome as shown in Fig. 7(a) and let $q = 3\,2\,1\,4\,8\,6\,7\,5 \in \mathfrak{S}_8$ be a circular permutation representing a circular genome as shown in Fig. 7(b). To find the minimum number of adjacent swaps needed to transform p into q is reduced to find the minimum number of adjacent swaps needed to sort circular permutation $q^{-1}p$. Since $q^{-1}p = 6\,3\,2\,4\,5\,8\,1\,7$, the problem is now to find the minimum number of adjacent swaps needed to sort circular permutation $6\,3\,2\,4\,5\,8\,1\,7$ (see Fig. 4), which was discussed in Section 3 (see Procedure 3.6).

The following specific cases of genomic rearrangement problems are directly relevant to the object sorting methods discussed in Section 3.

(i) Sorting by reversals of length 2 (i.e. adjacent swaps) on a linear (resp. circular) permutation [62,64–66]: Find a minimum-length sequence of reversals of length 2 (i.e. adjacent swaps) that sorts a permutation (resp. circular permutation) p.

(ii) Sorting by prefix-exchanges [14,67,68]: Find a minimum-length sequence of prefix-exchanges that sorts a permutation p, where a *prefix-exchange* [69] refers to an operation that swaps the first element of p

[2] Recall that a swap with swap distance 1 is also called an adjacent swap in this paper. The distinction is clear from context.

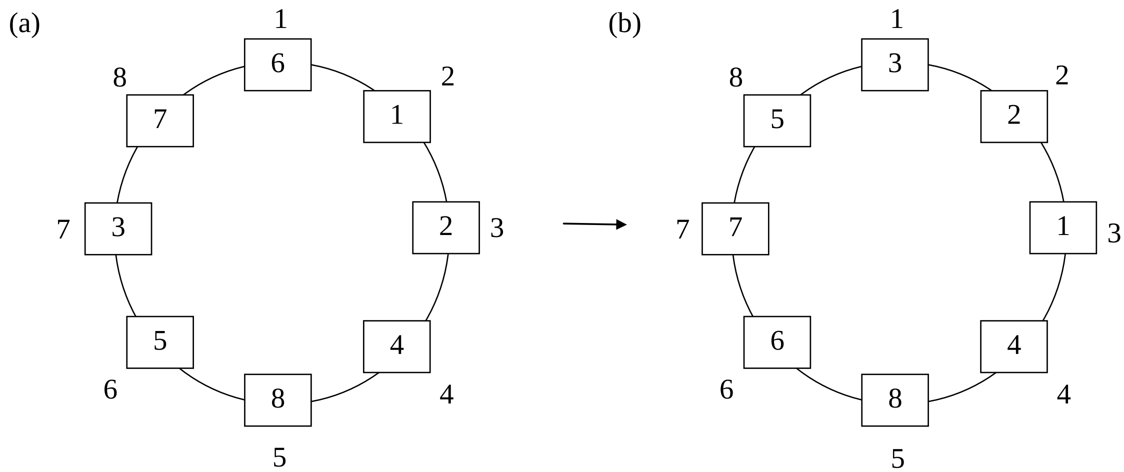

Fig. 7. Transforming a circular permutation representing a circular genome into another one.

with any other element of p.

(iii) Sorting by block-interchanges of length 1 (i.e. element-interchanges) [14, 67, 70]: Find a minimum-length sequence of block-interchanges of length 1 (i.e. element-interchanges) that sorts a permutation p.

Given a permutation p, (i) can be solved by using the bubble sort algorithm for a linear permutation and by using Procedure 3.6 for a circular permutation. (Sorting by reversals (or inversions) of length 2 (i.e. adjacent swaps) on a circular permutation can also be solved in polynomial time by lifting the problem to the problem involving the (*extended*) *affine symmetric group*. The interested reader may refer to [66] for further details.) Given a permutation p, (ii) can be solved by applying the permutation decomposition procedure shown in Procedure 3.2 with the input p^{-1}. Similarly, given a permutation p, (iii) can be solved by applying the permutation decomposition procedure shown in Procedure 3.4 with the input p^{-1}.

Now, consider that a permutation (resp. circular permutation) $p \in \mathfrak{S}_n$ representing a genome (resp. a circular genome). We see that sorting p by using the minimum number of reversals of length 2 corresponds to find a shortest path from permutation (resp. circular permutation) p to I in BS_n (resp. MBS_n).

Let d be a word metric on BS_n (i.e. Kendall's tau distance). The number of permutations $p \in \mathfrak{S}_n$ with $d(p, I) = k$ is exactly the number of permutations in $\mathfrak{S}_n$ with exactly k inversions, denoted by $T(n, k)$. The generating function for the numbers $T(n, k)$ is given as follows [4, 71]:

$$\prod_{i=1}^{n} \frac{1 - x^i}{1 - x} = \sum_{k=0}^{\infty} T(n, k) x^k. \tag{1}$$

In other words, the number of permutations in $\mathfrak{S}_n$ with exactly k inversions is the coefficient of x^k in (1).

Table 1. The number of permutations $p \in \mathfrak{S}_n$ with $d(p, I) = k, 1 \leq n \leq 8$, where d is a word metric on ST_n [72].

$n \backslash k$	0	1	2	3	4	5	6	7	8	9	10
1	1	0	0	0	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	0	0
3	1	2	2	1	0	0	0	0	0	0	0
4	1	3	6	9	5	0	0	0	0	0	0
5	1	4	12	30	44	26	3	0	0	0	0
6	1	5	20	70	170	250	169	35	0	0	0
7	1	6	30	135	460	1,110	1,689	1,254	340	15	0
8	1	7	42	231	1,015	3,430	8,379	13,083	10,048	3,409	315

Let $p \in \mathfrak{S}_n$ be a permutation representing a genome. Similarly, sorting p by using the minimum number of prefix-exchanges corresponds to find a shortest path from permutation p to I in ST_n. Now, let d be a word metric on ST_n. Table 1 shows the number of permutations $p \in \mathfrak{S}_n$ with $d(p, I) = k, 1 \leq n \leq 8$. The number of permutations $p \in \mathfrak{S}_n$ with $d(p, I) = k$ is exactly the *Whitney numbers of the second kind for the star poset* [72]. For $n \geq 1$ and $3 \leq k \leq \lfloor 3(n-1)/2 \rfloor$, it satisfies the following recurrence relation [73, 74]:

$$W_{n,k} = (n-1)W_{n-1,k-1} + \sum_{j=1}^{n-2} jW_{j,k-3}, \tag{2}$$

with $W_{n,0} = 1$, $W_{n,1} = n-1$, and $W_{n,2} = (n-1)(n-2)$. For example, we have $W_{4,3} = 9$, which means that there are nine permutations $q \in \mathfrak{S}_4$ with $d(q, I) = 3$ (see Table 1 with $n = 4$ and $k = 3$). For permutations $p \in \mathfrak{S}_n, 1 \leq n \leq 8$, the distribution of the distance involving prefix-exchanges is shown in Table 1.

Table 2. The number of permutations $p \in \mathfrak{S}_n$ with $d(p, I) = k, 1 \leq n \leq 10$, where d is a word metric on CT_n [14].

n \ k	0	1	2	3	4	5	6	7	8	9
1	1	0	0	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	0
3	1	3	2	0	0	0	0	0	0	0
4	1	6	11	6	0	0	0	0	0	0
5	1	10	35	50	24	0	0	0	0	0
6	1	15	85	225	274	120	0	0	0	0
7	1	21	175	735	1,624	1,764	720	0	0	0
8	1	28	322	1,960	6,769	13,132	13,068	5,040	0	0
9	1	36	546	4,536	22,449	67,284	118,124	109,584	40,320	0
10	1	45	870	9,450	63,273	269,325	723,680	1,172,700	1,026,576	362,880

Let $p \in \mathfrak{S}_n$ be a permutation representing a genome. We also see that sorting p by using the minimum number of block-interchanges of length 1 corresponds to find a shortest path from permutation p to I in CT_n. Now, let d be a word metric on CT_n. If permutation $p \in \mathfrak{S}_n$ is written as a product of disjoint cycles, then $d(p, I) = n - r$ in CT_n, where r is the number of disjoint cycles including fixed points [31]. Table 2 shows the number of permutations $p \in \mathfrak{S}_n$ with $d(p, I) = k, 1 \leq n \leq 10$. The number of permutations of n elements into m cycles is obtained by the *(unsigned) Stirling numbers of the first kind* [75], denoted by $s(n, m)$, for $1 \leq m \leq n$. It satisfies the following recurrence relation:

$$s(n, m) = s(n-1, m-1) + (n-1)s(n-1, m), \tag{3}$$

with $s(0, 0) = 1$ and $s(n, 0) = s(0, n) = 0$. For example, we have $s(3, 1) = 2$, which means that the number of permutations of three elements into one cycle is two, i.e., $(1\,2\,3) \in \mathfrak{S}_3$ and $(1\,3\,2) \in \mathfrak{S}_3$. It follows that there are two permutations $q \in \mathfrak{S}_3$ with $d(q, I) = 2$ (see Table 2 with $n = 3$ and $k = 2$). For permutations $p \in \mathfrak{S}_n, 1 \leq n \leq 10$, the distribution of the distance involving block-interchanges of length 1 is shown in Table 2.

6 Discussion and related works

In previous sections we focus on optimal sorting procedures by adjacent swaps. We discussed sorting n objects on n vertices in a graph of order n with a line, star, complete, or ring topology using the minimum number of adjacent swaps. When sorting objects at the vertices of graphs with some other topologies, we may consider a *k-approximation algorithm* [76] which guarantees that the output solution is no greater than k times optimal solution, where $k > 1$ for minimization problems. We first consider the problem of sorting n objects on n vertices in a tree T with n vertices by adjacent swaps. In [19] Akers and Krishnamurthy proposed a puzzle consisting of n markers on an arbitrary transposition tree with n vertices, which is basically the problem of sorting an initial arrangement of n objects on n vertices in a tree with n vertices by adjacent swaps. Some (not necessarily tight) upper bounds of the number of adjacent swaps needed to sort n objects on n vertices in a tree T with n vertices were studied by examining an upper bound of the diameter of the Cayley graph generated by the corresponding transposition tree [19, 77]. Let p be a permutation representing an initial arrangement of n objects on n vertices in an arbitrary tree T with n vertices, where both objects and vertices of T are numbered from 1 to n. We say that object i is *homed* if $i = p(i)$, where $p(i)$ denotes the i-th element of permutation p. Let $\eta(p)$ be the number of cycles in p and $d(i, j)$ be the distance between vertices i and j in T. The following procedure describes sorting the initial arrangement of n objects on n vertices in T by adjacent swaps, where its (not necessarily tight) upper bound of the number of adjacent swaps needed to sort the initial arrangement of n objects on n vertices in T is $\eta(p) - n + \Sigma_{i=1}^{n} d(i, p(i))$ (see Theorem 5 and appendix in [19]).

Procedure 6.1. SORTING OBJECTS ON A TREE BY ADJACENT SWAPS [19]
Input: A permutation $p \in \mathfrak{S}_n \, (n \geq 3)$ representing an initial bijective assignment between n objects (numbered from 1 to n) and n vertices (numbered from 1 to n) on a tree T with n vertices.
Output: A sequence of adjacent swaps needed to sort n objects in T.

- Initialize the sequence S.
- **while** $p \neq I$ **do**
  - Find an adjacent pair of vertices v_1 and v_2 such that their unhomed objects o_1 and o_2, respectively, are needed to move toward each other for their respective homed positions, i.e., $p(o_k) = o_k$ for $k = 1$ and $k = 2$. Or find an adjacent pair of vertices v_1 and v_2 such that object o_1 is homed while object o_2 is not homed such that object o_2 needs to move toward vertex v_1 for its homed position. Then, swap object o_1 on vertex v_1 and object o_2 on vertex v_2. Append transposition $(v_1 \, v_2)$ to the sequence S and update the permutation p reflecting the updated arrangement of n objects at the vertices of T.
- Return the sequence S.

In [10] Yamanaka et al. showed that token sorting involving n tokens and a graph of order n by adjacent swaps is solvable in $O(n^2)$ token swaps. They provide a polynomial-time 2α-approximation algorithm for the problem of finding the minimum number of adjacent swaps needed to sort an initial arrangement of n tokens on n vertices in a graph of order n whose *tree α-spanner* [78] can be computed in polynomial time.

Now, we consider sorting methods for graphs allowing swaps with swap distance at most $k \, (k \geq 2)$, which have not been discussed in previous sections. We see that the distance of each pair of vertices in a graph with a star topology is at most 2. We also see that the distance of each pair of vertices in a graph with a complete topology is 1. Therefore, it is easy to see that the minimum number of swaps with swap distance at most $k \, (k \geq 2)$ needed to sort an initial arrangement of n objects on n vertices, represented by permutation $p \in \mathfrak{S}_n$, in a graph of order n with a star topology is the same with the minimum number of swaps needed to sort the initial arrangement of n objects on n vertices, represented by the same permutation $p \in \mathfrak{S}_n$, in a graph of order n with a complete topology.

For a given graph G of order n with a line topology (resp. ring topology), the problem of finding the minimum number of swaps with swap distance at most $k \, (k \geq 2)$ needed to sort an initial arrangement of n objects on n vertices in G is basically the problem of finding the minimum number of $(k+1)$-bounded transpositions needed to sort a permutation (resp. a circular permutation), where a *k-bounded* $(k \geq 2)$ *transposition* [62] is an operation that swaps two elements having at most $k - 2$ elements in between. Although the solution of the above problem is not generally known, there are some approximation algorithms available, in particular for $k = 2$. (A 3-bounded transposition is called a *short swap* [62]. The interested reader may refer to [62, 79] for further details.)

As discussed in Section 5, permutation sorting plays a significant role in genomic rearrangements. Although we discussed polynomial-time algorithms for sorting linear or circular permutations by reversals of length 2 in Sections 3 and 5, sorting permutations by reversals is NP-hard in general [80], while sorting signed permutations by reversals is polynomial [81]. In [61] Solomon et al. proved that sorting circular permutations by reversals is also NP-hard.

Sorting by *prefix-reversals* [14, 82], which is also known as the *pancake sorting problem*, was studied on *pancake graphs* [82], where pancake graphs are Cayley graphs of symmetric groups with respect to the generating set of prefix-reversals. The exact diameter of an n-pancake graph is unknown in general, but it is known to lie between $17n/16$ and $(5n + 5)/3$ [83].

Sorting by transpositions used in genomic rearrangements is NP-hard [16], but there is a 1.375-approximation algorithm for sorting by transpositions presented by Elias and Hartman [53].

Sorting by block-interchanges is solvable in polynomial time [84]. Feng and Zhu [54] presented an $O(n \log n)$ solution for this problem using their data structure.

The solution spaces of genomic rearrangement problems have been researched extensively in recent years [56, 57, 85–88]. In [88] Siepel enumerated sorting solutions for sorting by reversals for high-level problems. The solution space of the All Sorting Reversals (ASR) problem is explored in [56, 57, 85, 86], which is used for sampling and estimating the total number of optimal sorting paths [89, 90]. In [85] Ajana et al. showed a method to allow the user to select one or several solutions for sorting signed permutations by reversals among the large number of minimal solutions.

To find multiple minimum-length sequences, if available, to sort $p \in \mathfrak{S}_n$ by adjacent swaps, we need to enumerate the reduced words of $p^{-1} \in \mathfrak{S}_n$ with respect to the generating set $S = \{(i\ i{+}1) : 1 \leq i < n\}$. This problem is researched in the area of combinatorics of words in Coxeter groups [36, 91]. Eriksson [92] proposed the *number game*, which provides a means of enumerating the reduced words of a permutation $q \in \mathfrak{S}_n$ with respect to the generating set $S = \{(i\ i{+}1) : 1 \leq i < n\}$ of $\mathfrak{S}_n$ using the theory of Coxeter groups.

To find multiple sequences, if available, to sort a permutation $p \in \mathfrak{S}_n$ by using the minimum number of prefix-exchanges, we need to enumerate minimum-length permutation decompositions (or factorizations) of $p^{-1} \in \mathfrak{S}_n$ using transpositions from the set $T = \{(1\ i) : 2 \leq i \leq n\}$. Permutation decompositions of a permutation $q \in \mathfrak{S}_n$ using the minimum number of transpositions from the set $T = \{(1\ i) : 2 \leq i \leq n\}$ is called *minimal star factorizations* [30] of $q \in \mathfrak{S}_n$. The enumeration of minimal star factorizations of a permutation is discussed in [93], while a formula for the total number of minimal star factorizations of a permutation is shown in [30].

In [94] Hurwitz found a formula for the number of *minimal transitive 2-cycle factorizations* [95] of a permutation with its cycle type. Bousquet-Mélou and Schaeffer [95] enumerated minimal transitive 2-cycle (i.e. transposition) factorizations of a permutation using planar constellations and Eulerian trees.

Meanwhile, in [11] Thompson and Kung presented an algorithm for sorting n^2 elements on an $n \times n$ mesh (or 2D) graph, where the swaps of their algorithm are not restricted to adjacent swaps. For a graph with a 2D grid (or 3D grid) topology, it is still open to find a (deterministic) polynomial time algorithm for finding a minimum-length sequence of adjacent swaps needed to sort n objects on n vertices in a graph of order n with a 2D grid (or 3D grid) topology.

7 Implementation

Permutation sorting and permutation decomposition procedures discussed in Section 3 run in (deterministic) polynomial time [4, 5, 18, 21, 69], which can be implemented as software modules. Although there are other utilities for permutation groups, such as GAP [96] and MAGMA [97], our implementation is specifically designed for procedures described in Section 3. The purpose of our implementation is to provide a tool for solving the adjacent swap-based object sorting problems discussed in Section 3 with the detailed output that allows the reader to follow each adjacent swap during an object sorting procedure. We now briefly describe our implementation, which is developed in the standard GNU C++ language [98] based on our previously implemented TSG tool [18]. The GNU C++ compiler is available free under the GNU General Public License (GPL) [99], which is included in most distributions of Linux platforms. It is also available for Windows platforms via a Cygwin toolkit [100], which provides a Linux-like environment for Windows platforms.

Our implementation had been tested both using gcc version 4.1.2 [98] running under a Linux platform and using gcc version 4.9.3 and Cygwin 2.3.1 running under a Windows platform. Source codes and data for our implementation are publicly available at the GitHub repository (see [101]), where GitHub is both an online code repository and a version control system [102]. See Table 3 for an example output of our implementation for a graph with a star topology.

Table 3. An example output for a graph with a star topology.

```
Graph Network Topology: Star Topology
Number of Objects: 9
Random Initial Source Permutation: No
Source Permutation (Initial Assignment): 7 9 2 1 6 4 5 3 8
Target Permutation: 1 2 3 4 5 6 7 8 9
Upper Bound (the diameter of star Cayley graph ST_n for n=9) : 12
```

The minimum number of adjacent swaps needed is 9.
<A permutation decomposition for the inverse of source permutation>
4 3 8 6 7 5 1 9 2 = (1 7)(1 5)(1 6)(1 4)(1 2)(1 9)(1 8)(1 3)(1 2)
Source Permutation: 7 9 2 1 6 4 5 3 8
1. After swapping objects between vertex 1 and vertex 7: 5 9 2 1 6 4 7 3 8
2. After swapping objects between vertex 1 and vertex 5: 6 9 2 1 5 4 7 3 8
3. After swapping objects between vertex 1 and vertex 6: 4 9 2 1 5 6 7 3 8
4. After swapping objects between vertex 1 and vertex 4: 1 9 2 4 5 6 7 3 8
5. After swapping objects between vertex 1 and vertex 2: 9 1 2 4 5 6 7 3 8
6. After swapping objects between vertex 1 and vertex 9: 8 1 2 4 5 6 7 3 9
7. After swapping objects between vertex 1 and vertex 8: 3 1 2 4 5 6 7 8 9
8. After swapping objects between vertex 1 and vertex 3: 2 1 3 4 5 6 7 8 9
9. After swapping objects between vertex 1 and vertex 2: 1 2 3 4 5 6 7 8 9
Target Permutation: 1 2 3 4 5 6 7 8 9
Upper Bound: 12
Reached the target permutation successfully (9 adjacent swaps).

Given a graph with a line, star, complete, or ring topology, our implementation takes permutation $p \in \mathfrak{S}_n$ $(n \geq 3)$ of n objects as an input for an initial bijective assignment between n objects and n vertices. Then, it shows a step-by-step output for sorting n objects by means of the minimum number of adjacent swaps using the generating set of $\mathfrak{S}_n$ corresponding to a given topology. It also shows an upper bound of the minimum number of adjacent swaps needed to sort n objects in a graph of order n with a given topology, which was discussed in Propositions 3.1, 3.3, 3.5, and 3.7.

The correctness of the permutation sorting and permutation decomposition procedures discussed in Section 3 was shown in the literature [4, 5, 18, 21, 30, 31, 69]. Our implementation basically implemented those proven procedures in order to provide a tool for solving the object sorting problems discussed in Section 3. One straightforward way to verify an output of our implementation is to first compute the minimum number of adjacent swaps needed to sort an initial arrangement of objects at the vertices of a graph with a given topology, then follow each (numbered) output step whether or not an adjacent swap has indeed been occurred. If the number of generated output steps coincides with the computed minimum number and each (numbered) output step is indeed an adjacent swap, the program has generated the desired output, which is a minimum-length sequence of adjacent swaps needed to sort the initial arrangement of objects at the vertices of the graph with the given topology. See Table 4 for another example output for a graph with a line topology.

Table 4. An example output for a graph with a line topology.

Graph Network Topology: Line Topology
Number of Objects: 10
Random Initial Source Permutation: Yes
Source Permutation (Initial Assignment): 1 9 2 7 10 6 4 3 8 5
Target Permutation: 1 2 3 4 5 6 7 8 9 10
Upper Bound (the diameter of bubble sort Cayley graph BS_n for n=10) : 45 (i.e.,nx(n-1)/2 for n=10).
The minimum number of adjacent swaps needed is 21.
<A permutation decomposition for the inverse of source permutation>
1 3 8 7 10 6 4 9 2 5 = (2 3)(3 4)(5 6)(6 7)(7 8)(8 9)(9 10)(4 5)(5 6)(6 7)(7 8)(8 9)(3 4)
(4 5)(5 6)(7 8)(3 4)(4 5)(6 7)(3 4)(5 6)
Source Permutation: 1 9 2 7 10 6 4 3 8 5
1. After swapping objects between vertex 2 and vertex 3 : 1 2 9 7 10 6 4 3 8 5
2. After swapping objects between vertex 3 and vertex 4 : 1 2 7 9 10 6 4 3 8 5
3. After swapping objects between vertex 5 and vertex 6 : 1 2 7 9 6 10 4 3 8 5
4. After swapping objects between vertex 6 and vertex 7 : 1 2 7 9 6 4 10 3 8 5
5. After swapping objects between vertex 7 and vertex 8 : 1 2 7 9 6 4 3 10 8 5
6. After swapping objects between vertex 8 and vertex 9 : 1 2 7 9 6 4 3 8 10 5
7. After swapping objects between vertex 9 and vertex 10 : 1 2 7 9 6 4 3 8 5 10
8. After swapping objects between vertex 4 and vertex 5 : 1 2 7 6 9 4 3 8 5 10
9. After swapping objects between vertex 5 and vertex 6 : 1 2 7 6 4 9 3 8 5 10
10. After swapping objects between vertex 6 and vertex 7 : 1 2 7 6 4 3 9 8 5 10
11. After swapping objects between vertex 7 and vertex 8 : 1 2 7 6 4 3 8 9 5 10
12. After swapping objects between vertex 8 and vertex 9 : 1 2 7 6 4 3 8 5 9 10
13. After swapping objects between vertex 3 and vertex 4 : 1 2 6 7 4 3 8 5 9 10

14. After swapping objects between vertex 4 and vertex 5 : 1 2 6 4 7 3 8 5 9 10
15. After swapping objects between vertex 5 and vertex 6 : 1 2 6 4 3 7 8 5 9 10
16. After swapping objects between vertex 7 and vertex 8 : 1 2 6 4 3 7 5 8 9 10
17. After swapping objects between vertex 3 and vertex 4 : 1 2 4 6 3 7 5 8 9 10
18. After swapping objects between vertex 4 and vertex 5 : 1 2 4 3 6 7 5 8 9 10
19. After swapping objects between vertex 6 and vertex 7 : 1 2 4 3 6 5 7 8 9 10
20. After swapping objects between vertex 3 and vertex 4 : 1 2 3 4 6 5 7 8 9 10
21. After swapping objects between vertex 5 and vertex 6 : 1 2 3 4 5 6 7 8 9 10
Target Permutation: 1 2 3 4 5 6 7 8 9 10
Upper Bound: 45
Reached the target permutation successfully (21 adjacent swaps).

In Table 4 we see that the minimum number of adjacent swaps needed to sort permutation 1 9 2 7 10 6 4 3 8 5 $\in \mathfrak{S}_{10}$ is 21, which can be computed by hand as well using its inversion number. The output generated by the program contains total 21 steps, where each (numbered) output step is indeed an adjacent swap.

In the “data” directory of our source codes at the GitHub repository [101] we included more complex types of input and output data for object sorting up to 200 objects on graphs with line, star, complete, and ring topologies. Since our implementation is freely available at a source code level under the free GPL license, the user may reuse our source codes for the user's specific implementation in order to apply them to other sorting problems.

8 Concluding remarks

This paper reviewed adjacent swap-based sorting methods for graphs with several well-known topologies. For a given permutation $p \in \mathfrak{S}_n$ representing an initial arrangement of n objects on n vertices in a graph of order n with a line, star, complete, or ring topology, the problem of finding a minimum-length sequence of adjacent swaps needed to sort n objects is reduced to find a minimum-length permutation decomposition of p^{-1} using the generating set of $\mathfrak{S}_n$ corresponding to a given topology. The known permutation sorting and permutation decomposition procedures were used to find a minimum-length sequence of adjacent swaps needed to sort objects on graphs with line, star, complete, and ring topologies. The known diameters of Cayley graphs were also used to find upper bounds of the minimum number of adjacent swaps needed to sort objects on those graphs.

We discussed several kinds of Cayley graphs as state diagrams for sorting objects on graphs with line, star, complete, and ring topologies. For example, the vertices of the bubble-sort graph BS_n (resp. modified bubble-sort graph MBS_n) represent the possible arrangements of n objects on n vertices in $L(n)$ (resp. $R(n)$), where $L(n)$ (resp. $R(n)$) denotes a graph of order n with a line topology (resp. ring topology). The edges of BS_n (resp. MBS_n) correspond to swapping objects on adjacent vertices in $L(n)$ (resp. $R(n)$). It follows that the distance between two vertices in BS_n (resp. MBS_n) represents the minimum number of adjacent swaps needed to reach from one arrangement of n objects to the other arrangement of n objects at the vertices of $L(n)$ (resp. $R(n)$).

We also discussed how adjacent swap-based sorting methods for graphs with several well-known topologies are applied to the combinatorics of genomic rearrangements. In particular, sorting a permutation (or circular permutation) by reversals of length 2, sorting by prefix-exchanges, and sorting by block-interchanges of length 1 can be simple applications of sorting objects on graphs with line, star, complete, and ring topologies by adjacent swaps.

To conclude, some possible future research directions are as follows:

- Although permutation sorting and permutation decomposition methods have been researched for the last several decades [4, 21, 26, 28, 30–32, 61, 62, 93], further researches are needed to consider sorting methods for graphs with more complex topologies (e.g. grid topology) and sorting methods for graphs with certain topologies (e.g. line and ring topology) using swaps with swap distance at most k ($k \geq 2$).

- More permutation group theory needs to be exploited to find a minimum-length generator sequence of a permutation with respect to a certain generating set and its generated permutation group. This is relevant, for example, to find a minimum-length sequence of adjacent swaps needed to sort objects on some specific types of trees, which are not discussed in this paper.

- The problem of enumerating optimal solutions and the problem of finding a closed formula for the total number of optimal solutions for an object sorting problem need also to be researched. They were solved only for the small number of cases, such as finding the total number of ways to sort an initial arrangement of n objects on n vertices in a graph of order n with a star topology by using the minimum number of adjacent swaps (see [30] for finding the total number of minimal star factorizations of a permutation).

- Finally, although we did not focus on approximation algorithms in this paper, the research on approximation algorithms for the problem of finding the minimum number of adjacent swaps (or swaps with swap distance at most $k\,(k \geq 2)$) needed to sort objects on some types of graphs (not discussed in this paper) can be a fruitful research direction. This research also includes finding an upper bound of the number of adjacent swaps (or swaps with swap distance at most $k\,(k \geq 2)$) needed to sort objects on those graphs.

References

[1] T. Hungerford, Algebra, Springer, New York, NY, 1980.

[2] P. Diaconis, Group representations in probability and statistics, Institute of Mathematical Statistics Lecture Notes—Monograph Series, Institute of Mathematical Statistics, Hayward, CA, 1988.

[3] M. Heydemann, Cayley Graphs and Interconnection Networks, in: G. Hahn, G. Sabidussi (Eds.), Graph Symmetry: Algebraic Methods and Applications, NATO ASI Series Volume 497, Springer, New York, NY, 1997, pp. 167–224.

[4] D. E. Knuth, The art of computer programming. Vol. 3: Sorting and searching, Addison-Wesley, Reading, MA, 1973.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to algorithms, 2nd Edition, The MIT Press, Cambridge, MA, 2001.

[6] K. Mehlhorn, Sorting and Searching (Eatcs Monographs on Theoretical Computer Science), Springer-Verlag New York, Inc., Secaucus, NJ, 1984.

[7] D. Graf, Scheduling and Sorting Algorithms for Robotic Warehousing Systems, Master's thesis, ETH Zürich, Zürich, Switzerland (2015).

[8] D. Graf, How to Sort by Walking on a Tree, in: N. Bansal, I. Finocchi (Eds.), Algorithms - ESA 2015: 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2015, pp. 643–655.

[9] F.-C. Lin, J. Y.-J. Hsu, A genetic algorithm approach for the object-sorting task problem, in: Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on, Vol. 1, IEEE, New York, NY, 1995, pp. 241–246.

[10] K. Yamanaka, E. D. Demaine, T. Ito, J. Kawahara, M. Kiyomi, Y. Okamoto, T. Saitoh, A. Suzuki, K. Uchizawa, T. Uno, Swapping labeled tokens on graphs, Theoretical Computer Science 586 (2015) 81 – 94.

[11] C. D. Thompson, H. T. Kung, Sorting on a Mesh-connected Parallel Computer, in: Proceedings of the Eighth Annual ACM Symposium on Theory of Computing, STOC '76, ACM, New York, NY, 1976, pp. 58–64.

[12] S. Rajasekaran, D. S. Wei, Selection, routing, and sorting on the star graph, J. Parallel Distrib. Comput. 41 (2) (1997) 225–233.

[13] L. S. Heath, J. P. C. Vergara, Sorting by bounded block-moves, Discrete Applied Mathematics 88 (1998) 181–206.

[14] G. Fertin, A. Labarre, I. Rusu, É. Tannier, S. Vialette, Combinatorics of Genome Rearrangements, MIT Press, Cambridge, MA, 2009.

[15] D. A. Christie, Genome rearrangement problems, Ph.D. thesis, University of Glasgow, Glasgow, UK (1998).

[16] L. Bulteau, G. Fertin, I. Rusu, Sorting by Transpositions is Difficult, SIAM Journal on Discrete Mathematics 26 (3) (2012) 1148–1180.

[17] K. Yamanaka, T. Horiyama, D. Kirkpatrick, Y. Otachi, T. Saitoh, R. Uehara, Y. Uno, Swapping Colored Tokens on Graphs, in: F. Dehne, J.-R. Sack, U. Stege (Eds.), Algorithms and Data Structures: 14th International Symposium, WADS 2015, Victoria, BC, Canada, August 5-7, 2015. Proceedings, Springer International Publishing, Cham (ZG), Switzerland, 2015, pp. 619–628.

[18] D. Kim, Task swapping networks in distributed systems, International Journal of Computer Mathematics 90 (2013) 2221–2243.

[19] S. B. Akers, B. Krishnamurthy, A Group-Theoretic Model for Symmetric Interconnection Networks, IEEE Transactions on Computers 38 (1989) 555–566.

[20] S. Lakshmivarahan, J.-S. Jwo, S. K. Dhall, Symmetry in interconnection networks based on Cayley graphs of permutation groups: A survey, Parallel Computing 19 (4) (1993) 361–407.

[21] M. Jerrum, The complexity of finding minimum-length generator sequences, Theoretical Computer Science 36 (1985) 265–289.

[22] J. B. Fraleigh, A First Course in Abstract Algebra, Addison-Wesley, Reading, MA, 1998.

[23] C. K. Enyioha, D. C. Tarraf, L. Li, J. C. Doyle, On the graph of trees, in: IEEE CCA/ISIC 2009, Saint Petersburg, Russia, 2009, pp. 246–248.

[24] T. Enokido, M. Takizawa, Group Communication Protocol for Autonomic Computing, in: International Conference on Parallel and Distributed Systems, Fukuoka, 2005, pp. 443–447.

[25] H. L. Zhang, C. H. C. Leung, G. K. Raikundalia, Classification of Intelligent Agent Network Topologies and a New Topological Description Language for Agent Networks, in: Z. Shi, K. Shimohara, D. Feng (Eds.), Intelligent Information Processing III, Vol. 228 of IFIP International Federation for Information Processing, 2006, pp. 21–31.

[26] A. Garsia, The Saga of Reduced Factorizations of Elements of the Symmetric Group, Publications du LaCIM 29, Université du Québec á Montréal (2002).

[27] Q. Zhu, Topologies of agents interactions in knowledge intensive multi-agent systems for networked information services, Advanced Engineering Informatics 20 (2006) 31–45.

[28] I. Pak, Reduced decompositions of permutations in terms of star transpositions, generalized Catalan numbers and k-ARY trees, Discrete Mathematics 204 (1999) 329–335.

[29] A. van Zuylen, J. Bieron, F. Schalekamp, G. Yu, An upper bound on the number of circular transpositions to sort a permutation, arXiv:1402.4867 (2014).

[30] J. Irving, A. Rattan, Factorizations of Permutations into Star Transpositions, Discrete Mathematics 309 (2009) 1435–1442.

[31] G. Mackiw, Permutations as Products of Transpositions, The American Mathematical Monthly 102 (1995) 438–440.

[32] D. Neuenschwander, On the Representation of Permutations as Products of Transpositions, Elemente der Mathematik 56 (2001) 1–3.

[33] N. Biggs, Algebraic Graph Theory, Cambridge University Press, Cambridge, MA, 2013.

[34] L. Babai, Spectra of Cayley graphs, Journal of Combinatorial Theory, Series B 27 (2) (1979) 180 – 189.

[35] J. Meier, Groups, Graphs and Trees: An Introduction to the Geometry of Infinite Groups, Cambridge University Press, Cambridge, MA, 2008.

[36] A. Björner, F. Brenti, Combinatorics of Coxeter Groups, Springer, New York, NY, 2005.

[37] M. Camelo, P. Vilà, L. Fàbrega, D. Papadimitriou, Cayley-Graph-Based Data Centers and Space Requirements of a Routing Scheme Using Automata, in: Proceedings of the 2014 IEEE 34th International Conference on Distributed Computing Systems Workshops, ICDCSW '14, IEEE Computer Society, Washington, DC, 2014, pp. 63–69.

[38] M. Camelo, D. Papadimitriou, L. Fàbrega, P. Vilà, Efficient routing in data center with underlying cayley graph, in: P. Contucci, R. Menezes, A. Omicini, J. Poncela-Casasnovas (Eds.), Complex Networks V: Proceedings of the 5th Workshop on Complex Networks CompleNet 2014, Springer International Publishing, Cham (ZG), Switzerland, 2014, pp. 189–197.

[39] W. J. Gutjahr, M. Hitz, T. A. Mueck, Task assignment in Cayley interconnection topologies, Parallel Computing 23 (10) (1997) 1429–1460.

[40] P. Boldi, S. Vigna, Minimal Sense of Direction and Decision Problems for Cayley Graphs, Inf. Process. Lett. 64 (6) (1997) 299–303.

[41] D. Kim, E. Noel, K. W. Tang, Expanded Borel Cayley Graphs (Ex-BCGs): A novel communication topology for multi-agent systems, Journal of Network and Computer Applications 37 (2014) 47–61.

[42] T. Schiavinotto, T. Stützle, A review of metrics on permutations for search landscape analysis, Computers & Operations Research 34 (10) (2007) 3143–3153.

[43] E. Cheng, L. Lipták, N. Shawash, Orienting Cayley graphs generated by transposition trees, Computers and Mathematics with Applications 55 (11) (2008) 2662–2672.

[44] K. Kaneko, Y. Suzuki, Node-to-node internally disjoint paths problem in bubble-sort graphs, in: Dependable Computing, 2004. Proceedings. 10th IEEE Pacific Rim International Symposium on, IEEE, New York, NY, 2004, pp. 173–182.

[45] K. Qiu, S. G. Akl, On Node-to-Node Disjoint Paths in the Star Interconnection Network, in: International Conference on Parallel and Distributed Computing Systems, PDCS 2005, November 14–16, IASTED, Calgary, Alberta, Canada, 2005, pp. 731–735.

[46] Y. Suzuki, K. Kaneko, M. Nakamori, Node-Disjoint Paths Algorithm in a Transposition Graph, IEICE - Trans. Inf. Syst. E89-D (10) (2006) 2600–2605.

[47] M. Dietzfelbinger, S. Madhavapeddy, I. H. Sudborough, Three disjoint path paradigms in star networks, in: Parallel and Distributed Processing, 1991. Proceedings of the Third IEEE Symposium on, Dec 2–5, IEEE, New York, NY, 1991, pp. 400–406.

[48] C.-N. Lai, On the construction of all shortest vertex-disjoint paths in Cayley graphs of abelian groups, Theoretical Computer Science 571 (2015) 10–20.

[49] C. N. Lai, Optimal Construction of All Shortest Node-Disjoint Paths in Hypercubes with Applications, IEEE Transactions on Parallel and Distributed Systems 23 (6) (2012) 1129–1134.

[50] K. Kaneko, N. Sawada, An algorithm for node-to-node disjoint paths problem in burnt pancake graphs, IEICE - Trans. Inf. Syst. E90-D (1) (2007) 306–313.

[51] M. Snyder, M. Gerstein, Defining Genes in the Genomics Era, Science 300 (5617) (2003) 258–260.

[52] N. C. Jones, P. Pevzner, An introduction to bioinformatics algorithms, MIT Press, Cambridge, MA, 2004.

[53] I. Elias, T. Hartman, A 1.375-Approximation Algorithm for Sorting by Transpositions, IEEE/ACM Transactions on Computational Biology and Bioinformatics 3 (4) (2006) 369–379.

[54] J. Feng, D. Zhu, Faster Algorithms for Sorting by Transpositions and Sorting by Block Interchanges, ACM Trans. Algorithms 3 (3) (2007) Article No. 25.

[55] V. Bafna, P. A. Pevzner, Genome rearrangements and sorting by reversals, SIAM J. Comput. 25 (2) (1996) 272–289.

[56] M. D. Braga, M.-F. Sagot, C. Scornavacca, E. Tannier, Exploring the Solution Space of Sorting by Reversals, with Experiments and an Application to Evolution, IEEE/ACM Transactions on Computational Biology and Bioinformatics 5 (3) (2008) 348–356.

[57] G. Badr, K. M. Swenson, D. Sankoff, Listing All Parsimonious Reversal Sequences: New Algorithms and Perspectives, in: Proceedings of the 2010 International Conference on Comparative Genomics, RECOMB-CG'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 39–49.

[58] Y. C. Lin, C. L. Lu, H.-Y. Chang, C. Y. Tang, An Efficient Algorithm for Sorting by Block-Interchanges and Its Application to the Evolution of Vibrio Species, Journal of Computational Biology 12 (2005) 102–112.

[59] L. L. Chin, C. L. Ying, L. H. Yen, Y. T. Chuan, Comparative Genomics, Humana Press, Totowa, NJ, 2007, Ch. Analysis of Genome Rearrangement by Block-Interchanges, pp. 121–134.

[60] C. St. Clair, J. E. Visick, Exploring Bioinformatics: A Project-Based Approach, 2nd Edition, Jones and Bartlett Publishers, Inc., Sudbury, MA, 2013.

[61] A. Solomon, P. Sutcliffe, R. Lister, Sorting circular permutations by reversal, in: F. Dehne, J.-R. Sack, M. Smid (Eds.), Algorithms and Data Structures: 8th International Workshop, WADS 2003, Ottawa, Ontario, Canada, July 30 - August 1, 2003. Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, pp. 319–328.

[62] X. Feng, B. Chitturi, H. Sudborough, Sorting Circular Permutations by Bounded Transpositions, in: H. R. Arabnia (Ed.), Advances in Computational Biology, Vol. 680 of Advances in Experimental Medicine and Biology, Springer, New York, NY, 2010, pp. 725–736.

[63] B. Chitturi, H. Sudborough, W. Voit, X. Feng, Adjacent swaps on strings, in: Proceedings of the 14th Annual International Conference on Computing and Combinatorics, COCOON '08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 299–308.

[64] P. A. Pevzner, Computational Molecular Biology: An Algorithmic Approach, MIT Press, Cambridge, MA, 2000.

[65] V. Bafna, D. Beaver, M. Fürer, P. A. Pevzner, Circular Permutations and Genome Shuffling, in: D. Sankoff, J. H. Nadeau (Eds.), Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment and the Evolution of Gene Families, Springer Netherlands, Dordrecht, Netherlands, 2000, pp. 199–206.

[66] A. Egri-Nagy, V. Gebhardt, M. M. Tanaka, A. R. Francis, Group-theoretic models of the inversion process in bacterial genomes, Journal of Mathematical Biology 69 (1) (2014) 243–265.

[67] A. Labarre, Edit distances and factorisations of even permutations, in: Proceedings of the 16th Annual European Symposium on Algorithms, ESA '08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 635–646.

[68] A. Labarre, Lower Bounding Edit Distances between Permutations, SIAM Journal on Discrete Mathematics 27 (3) (2013) 1410–1428.

[69] S. B. Akers, D. Harel, B. Krishnamurthy, The star graph: An attractive alternative to the n-cube, in: I. D. Scherson, A. S. Youssef (Eds.), Interconnection Networks for High-performance Parallel Computers, IEEE Computer Society Press, Los Alamitos, CA, 1994, pp. 145–152.

[70] T. A. de Lima, M. Ayala-Rincn, Complexity of Cayley distance and other general metrics on permutation groups, in: Computing Congress (CCC), 2012 7th Colombian, 2012, pp. 1–6.

[71] A. Barg, A. Mazumdar, Codes in Permutations and Error Correction for Rank Modulation, IEEE Transactions on Information Theory 56 (7) (2010) 3158–3165.

[72] F. J. Portier, T. P. Vaughan, Whitney Numbers of the Second Kind for the Star Poset, European Journal of Combinatorics 11 (3) (1990) 277–288.

[73] K. Qiu, S. G. Akl, On Some Properties of the Star Graph, VLSI Design 2 (4) (1995) 389–396.

[74] S. Grusea, A. Labarre, Asymptotic normality and combinatorial aspects of the prefix exchange distance distribution, Advances in Applied Mathematics 78 (2016) 94–113.

[75] J. Arndt, Matters Computational: Ideas, Algorithms, Source Code, 1st Edition, Springer-Verlag New York, Inc., New York, NY, 2010.

[76] D. P. Williamson, D. B. Shmoys, The Design of Approximation Algorithms, Cambridge University Press, New York, NY, 2011.

[77] A. Ganesan, An efficient algorithm for the diameter of Cayley graphs generated by transposition trees, IAENG International Journal of Applied Mathematics 42 (4) (2012) 214–223.

[78] L. Cai, D. G. Corneil, Tree Spanners, SIAM J. Discret. Math. 8 (3) (1995) 359–387.

[79] L. S. Heath, J. C. Vergara, Sorting by short swaps, Journal of Computational Biology 10 (2003) 775–789.

[80] A. Caprara, The Reversal Median Problem, INFORMS J. on Computing 15 (1) (2003) 93–113.

[81] S. Hannenhalli, P. A. Pevzner, Transforming Cabbage into Turnip: Polynomial Algorithm for Sorting Signed Permutations by Reversals, J. ACM 46 (1) (1999) 1–27.

[82] E. Konstantinova, Information Theory, Combinatorics, and Search Theory: In Memory of Rudolf Ahlswede, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, Ch. On Some Structural Properties of Star and Pancake Graphs, pp. 472–487.

[83] W. H. Gates, C. H. Papadimitriou, Bounds for sorting by prefix reversal, Discrete Mathematics 27 (1) (1979) 47–57.

[84] D. A. Christie, Sorting permutations by block-interchanges, Information Processing Letters 60 (4) (1996) 165 – 169.

[85] Y. Ajana, L. Jean-François, E. R. Tillier, N. El-Mabrouk, Exploring the Set of All Minimal Sequences of Reversals – An Application to Test the Replication-Directed Reversal Hypothesis, in: R. Guigó, D. Gusfield (Eds.), Algorithms in Bioinformatics: Second International Workshop, WABI 2002 Rome, Italy, September 17–21, 2002 Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002, pp. 300–315.

[86] Amritanjali, G. Sahoo, Listing sorting sequences of reversals and translocations, in: Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics, BCB'13, ACM, New York, NY, p. 712.

[87] A. Ouangraoua, A. Bergeron, Combinatorial Structure of Genome Rearrangements Scenarios., Journal of Computational Biology 17 (2010) 1129–1144.

[88] A. C. Siepel, An Algorithm to Enumerate All Sorting Reversals, in: Proceedings of the Sixth Annual International Conference on Computational Biology, RECOMB '02, ACM, New York, NY, 2002, pp. 281–290.

[89] I. Miklós, A. Darling, Efficient Sampling of Parsimonious Inversion Histories with Application to Genome Rearrangement in Yersinia, Genome Biology and Evolution 1 (1) (2009) 153–164.

[90] I. Miklós, H. Smith, Sampling and counting genome rearrangement scenarios, BMC Bioinformatics 16 (14) (2015) 1–14.

[91] R. P. Stanley, On the Number of Reduced Decompositions of Elements of Coxeter Groups, European Journal of Combinatorics 5 (4) (1984) 359–372.

[92] H. Eriksson, Computational and combinatorial aspects of Coxeter groups, Ph.D. thesis, KTH Royal Institute of Technology, Stockholm, Sweden (1994).

[93] B. E. Tenner, A combinatorial proof of symmetry among minimal star factorizations, Discrete Mathematics 312 (16) (2012) 2482–2490.

[94] A. Hurwitz, Ueber Riemann'sche Flächen mit gegebenen Verzweigungspunkten, Mathematische Annalen 39 (1) (1891) 1–60. doi:10.1007/BF01199469.

[95] M. Bousquet-Mlou, G. Schaeffer, Enumeration of Planar Constellations, Advances in Applied Mathematics 24 (4) (2000) 337 – 368.

[96] The GAP Group, GAP - Groups, Algorithms, Programming, Version 4.8.3, http://www.gap-system.org (2016).

[97] W. Bosma, J. Cannon, C. Playoust, The Magma Algebra System I: The User Language, Journal of Symbolic Computation 24 (3) (1997) 235–265.

[98] Free Software Foundation, The GNU C++ Library (libstdc++ documentation), http://gcc.gnu.org/onlinedocs/libstdc++.

[99] F. S. Foundation, GNU General Public License, Version 3, http://www.fsf.org/copyleft/gpl.html (2007).

[100] J. Racine, The Cygwin tools: a GNU toolkit for Windows, Journal of Applied Econometrics 15 (3) (2000) 331–341.

[101] D. Kim, The AdjacentSwaps Package, Version 1.0, https://github.com/SortingOnGraphs/AdjacentSwaps (2016).

[102] J. D. Blischak, E. R. Davenport, G. Wilson, A Quick Introduction to Version Control with Git and GitHub, PLoS Computational Biology 12 (1) (2016) e1004668. doi:10.1371/journal.pcbi.1004668.