Generalized Gene Adjacencies, Graph Bandwidth, and Clusters in Yeast Evolution

Qian Zhu, Zaky Adam, Vicky Choi, and David Sankoff

Abstract—We present a parameterized definition of gene clusters that allows us to control the emphasis placed on conserved order within a cluster. Though motivated by biological rather than mathematical considerations, this parameter turns out to be closely related to the bandwidth parameter of a graph. Our focus will be on how this parameter affects the characteristics of clusters: how numerous they are, how large they are, how rearranged they are, and to what extent they are preserved from ancestor to descendant in a phylogenetic tree. We infer the latter property by dynamic programming optimization of the presence of individual edges at the ancestral nodes of the phylogeny. We apply our analysis to a set of genomes drawn from the Yeast Gene Order Browser.

Index Terms—Comparative genomics, gene clusters, yeast, evolution, phylogeny, genome rearrangements, graph bandwidth, dynamic programming, Saccharomyces cerevisiae, Candida glabrata, Ashbya gossypii, Kluyveromyces waltii, Kluyveromyces lactis.

1 Introduction

The definition of synteny blocks, gene clusters, or similar constructs from the comparison of two or more genomes entails a trade-off of great consequence: if we place emphasis on identical content and order of the genes, segments, or markers in a block or cluster, only relatively small regions of the genome will satisfy this restrictive condition, giving rise to a plethora of tiny blocks while missing large regions common to the genomes. On the other hand, by allowing unrestricted scrambling of genes within blocks (e.g., max-gap [1] or "gene teams" [7]), we forgo accounting for local genome rearrangement, missing an important aspect of evolutionary history, or we relinquish the possibility of pinpointing extensive local conservation, where this exists.

In this paper, we present a parameterized definition of gene clusters that allows us to control the emphasis placed on conserved order within a cluster. Though motivated by biological rather than mathematical considerations, this parameter turns out to be closely related to the bandwidth parameter of a graph. Our focus will be on how this parameter affects the characteristics of clusters: how numerous they are, how large they are, how rearranged they are, and to what extent they are preserved from ancestor to descendant in a phylogenetic tree. We infer the latter property by dynamic programming optimization of the presence of individual edges in a generalized adjacency (GA) graph abstractly representing chromosomal gene order. We apply our analysis to a set of genomes drawn from the Yeast Gene Order Browser (YGOB) [2]. Among the results, we find strong evidence for setting a certain fixed value to the cluster parameter. We also find that we can recover almost all the clusters that can be found without order constraints, i.e., by the max-gap criterion, indicating that local order conservation is a lot greater than that unconstrained definition would suggest.

2 Definitions and Preliminaries

Our characterization of gene clusters is made up of a general part that identifies clusters of vertices common to two graphs, and a specific part where a graph is determined by the proximity of genes on the chromosomes of a genome. This is illustrated in Fig. 1.

Definition 1. *Let $G_S = (V_S, E_S)$ and $G_T = (V_T, E_T)$ be two graphs with a nonempty set of vertices in common $V = V_S \cap V_T$. We say that a subset of $C \subseteq V$ is a* ***GA cluster*** *if it is the vertex set of a connected component of $G_{ST} = (V, E_S \cap E_T)$.*

Definition 2. *For the purposes of genome comparison, we may consider V_X to be the set of genes in the genome X. For genes g and h in V_X on the same chromosome in X, let $gh \in E_X$ if the number of genes intervening between g and h in X is less than θ, where $\theta \geq 1$ is a fixed* ***neighborhood parameter****.*

These definitions of edge sets and GA clusters decompose the genes in the two genomes into identical sets of disjoint clusters of size greater than or equal to 2, and possibly different sets of singletons belonging to no cluster, either because they are in V, but not adjacent to an edge in $E_S \cap E_T$, or because they are in $(V_S \cup V_T \setminus V)$. For simplicity, we do not attempt to deal with duplicate genes in this paper. When $\theta = 1$, a cluster has exactly the same gene content and order (or reversed order) in both genomes.

- Q. Zhu is with the Department of Biochemistry, University of Ottawa, 550 Cumberland St., Ottawa, ON K1N 6N5, Canada. E-mail: qzhu012@uottawa.ca.
- Z. Adam is with the School of Information Technology and Engineering, University of Ottawa, 550 Cumberland St., Ottawa, ON K1N 6N5, Canada. E-mail: zadam008@uottawa.ca.
- V. Choi is with the Department of Computer Science, Virginia Tech., 2050 Torgersen Hall, Blacksburg, VA 24061. E-mail: vchoi@cs.vt.edu.
- D. Sankoff is with the Department of Mathematics, University of Ottawa, 550 Cumberland St., Ottawa, ON K1N 6N5, Canada. E-mail: sankoff@uottawa.ca.

Manuscript received 23 June 2008; revised 13 Oct. 2008; accepted 22 Oct. 2008; published online 4 Nov. 2008.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBBSI-2008-06-0117.
Digital Object Identifier no 10.1109/TCBB.2008.121.

1545-5963/09/$25.00 © 2009 IEEE Published by the IEEE CS, CI, and EMB Societies & the ACM

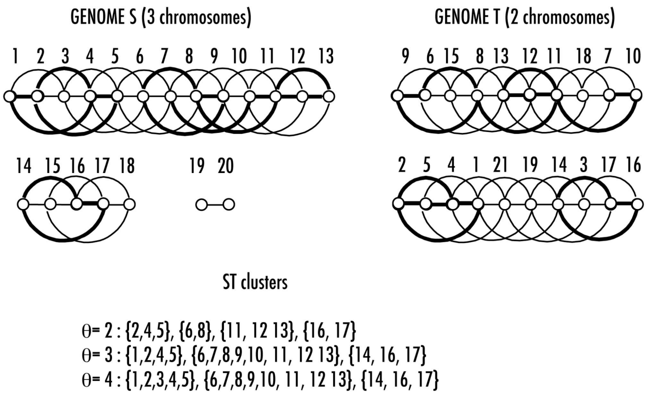

GENOME S (3 chromosomes)
1 2 3 4 5 6 7 8 9 10 11 12 13
14 15 16 17 18
19 20
GENOME T (2 chromosomes)
9 6 15 8 13 12 11 18 7 10
2 5 4 1 21 19 14 3 17 16
ST clusters
θ= 2 : {2,4,5}, {6,8}, {11, 12 13}, {16, 17}
θ= 3 : {1,2,4,5}, {6,7,8,9,10, 11, 12 13}, {14, 16, 17}
θ= 4 : {1,2,3,4,5}, {6,7,8,9,10, 11, 12 13}, {14, 16, 17}

Fig. 1. Graphs constructed from two genomes using parameter $\theta = 3$. Thick edges determine clusters. GA clusters are listed for $\theta = 2$ and $\theta = 4$ as well.

When $\theta = \infty$, the definition returns simply all the synteny sets, namely the sets of genes in common between two chromosomes, one in each genome.

Let Π be the set of all orderings of V. Recall that the **bandwidth** of a graph $G(V, E)$ is defined to be

$$B(G) = \min_{p \in \Pi} \max_{uv \in E} |p(u) - p(v)|. \tag{1}$$

In a genome S, each chromosome χ determines a physical order among the genes it contains.

Proposition 1. *Bandwidth $B(G_S) = \theta$, as long as there are at least $2\theta + 1$ genes on some chromosome χ in genome S.*

Proof. By Definition 2, the vertex v corresponding to the gene at position $\theta + 1$ on chromosome χ is connected to 2θ other vertices. The most remote of these are at positions 1 and $2\theta + 1$. In general, for a vertex u at any other position on χ, we can show that the most remote gene connected to u is no farther away than θ. Thus, for the order $p(\cdot)$ on the vertices defined by the original gene order, $\max|p(u) - p(v)| = \theta$. Hence, $B(G_S) \leq \theta$.

For any other order $p(\cdot)$, consider the 2θ vertices connected to vertex v. For one such vertex w, $|p(v) - p(w)| \geq \theta$, since we cannot fit 2θ vertices connected to v into an interval of size $< 2\theta + 1$, also containing v.

Since the upper and lower bounds coincide, the proposition follows. □

Proposition 2.

$$B(G_{ST}) = \max_{C \in \mathcal{C}} B(C), \tag{2}$$

where $\mathcal{C}$ is the set of connected components of G_{ST}.

Proof. Since E_{ST} is the union of the edges in all C

$$\max_{uv \in E_{ST}} |p(u) - p(v)| = \max_{C \in \mathcal{C}} \max_{uv \in E_C} |\bar{p}(u) - \bar{p}(v)|, \tag{3}$$

where $\bar{p}(\cdot)$ is the order induced on the vertices in C by the order $p(\cdot)$ on E_{ST}. But any set of vertex orders on all the individual C can be jointly extended to an order on V_{ST}. □

We compare the definition of a GA cluster with that of a **max-gap cluster** [7], [1], briefly reiterated below.

Definition 3. *Let $\theta \geq 1$. Let $V_C \subseteq V_S \cap V_T$ be a set of r vertices corresponding to genes all on the same chromosome χ_S in genome S and all on the same chromosome χ_T in genome T. Let $g_1, g_2, \ldots, g_r$ be a labeling of these genes according to their order on χ_S. Let $h_1, h_2, \ldots, h_r$ be a labeling of these same genes according to their order on χ_T. Let $p_S(\cdot)$ and $p_T(\cdot)$ indicate the positions of genes on χ_S and χ_T, respectively. Then, if*

$$p_S(g_{i+1}) - p_S(g_i) \leq \theta \text{ and } p_T(h_{j+1}) - p_T(h_j) \leq \theta, \tag{4}$$

for all $1 \leq i, j \leq r - 1$, then V_C satisfies the max-gap criterion. If in addition, V_C is contained in no larger V_F also satisfying the criterion, then V_C is said to be a max-gap cluster.

Proposition 3. *Every GA cluster with parameter θ satisfies the max-gap criterion with the same value of θ.*

Proof. Consider two successive genes in the GA cluster in genome S. By Definition 2, they cannot be separated by more than $\theta - 1$ genes not in the cluster. Since this holds for all pairs of successive genes, both in S and in T, the max-gap criterion is met. □

The converse of Proposition 3 does not hold, however. The max-gap criterion limits only the number of **noncluster elements** intervening, in either genome, between two cluster elements. Thus, in the max-gap definition with $\theta = 2$, we could have a cluster $\{a, b, c, d, e, f\}$ with order $a * bcdef$ in S and $fbdce * a$ in T, where the asterisks represent genes not present, or remote, in one of S or T, but this would not be a GA cluster (though $\{b, c, d, e\}$ would be). Condition (4) holds for both S and T since $p_S(g_2) - p_S(g_1) = 2$, $p_S(h_6) - p_S(h_5) = 2$, while the remaining $p_S(g_{i+1}) - p_S(g_i) = 1$ and the remaining $p_T(h_{i+1}) - p_T(h_i) = 1$. Because there is no edge in $E_S \cap E_T$ incident to vertex a or f, neither vertex can be in GA cluster.

Both max-gap and GA criteria have been analyzed for the purposes of statistically testing clusters against the null hypothesis that genomes S and T are randomized with respect to each other [7], [11].

Note that it is easy to identify GA clusters since graphs like those in Fig. 1 are trivial to construct, as is the intersection of the edge sets. The identification of connected components in a graph is a standard linear-time algorithm.

3 Comparisons of Yeast Genomes

3.1 The Data

The YGOB [2] contains complete gene orders and orthology identification among the five yeast species depicted in Fig. 2: two descendents of an ancient genome duplication event, *Saccharomyces cerevisiae* and *Candida glabrata*, and three species that diverged before this event, *Ashbya gossypii, Kluyveromyces waltii,* and *Kluyveromyces lactis.* For the ancient tetraploids, YGOB includes a reconstruction of the ancestral genome, which, with the help of further details supplied by Kevin Byrne and Jonathan Gordon (personal communication), allows us to identify duplicate genes as belonging to one of the two ancestral lineages, indicated by A and B in the figure, and to find two complete sets of clusters in each of these

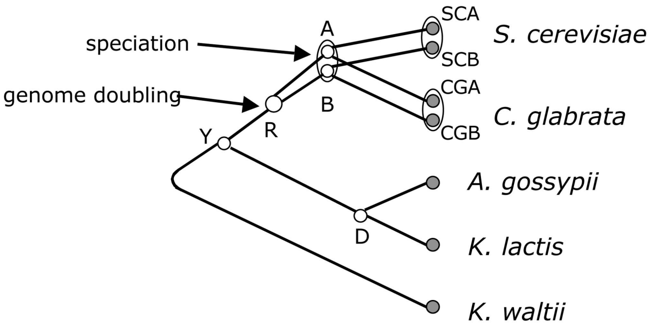


Fig. 2. Phylogeny of yeasts in YGOB. Whole genome doubling event at R giving rise to A and B lineages in *S. cerevisiae* (SCA, SCB) and *C. glabrata* (CGA, CGB) indicated, as is the speciation event at the divergence of these two species. Choice among the identified ancestor nodes Y, R, D, A, or B to be the root is arbitrary in our mathematical analysis, but historically, the earliest divergence time is represented by the branching at the left of the phylogeny.

species, one in each lineage. For our purposes, then, the duplicate lineage effectively expands the data set from five to seven genomes.

3.2 Notation

With reference to Fig. 2, we will refer to the common ancestor of *Ashbya gossypii* and *Kluyveromyces lactis* as Node D, and to its immediate ancestor as Y. Nodes A and B will refer to the two ancestral lineages within both *Saccharomyces cerevisiae* and *Candida glabrata* at the time of speciation, while Node R will designate the tetraploid ancestral to these.

3.3 GA Clusters of Diploid Genomes and Comparison with Max-Gap Criterion

We constructed GA clusters between every pair of diploids chosen from *A. gossypii*, *K. lactis*, and *K. waltii*. Fig. 3 reveals that *K. waltii* returns fewer and larger clusters than the other diploids, as we would expect from its closer relationship to the diploid ancestor Y. Additionally, the number of clusters detected as a function of θ, decreases as a result of cluster amalgamation, featuring a distinct elbow near $\theta = 3$ for all the pairwise comparisons. This also shows a striking resemblance to the same analysis for max-gap clusters, suggesting that in these data, the max-gap clusters also satisfy our more stringent GA criterion. In other contexts, perhaps in prokaryotes, more intense processes of local gene rearrangement may result in relatively more max-gap clusters.

3.4 Defining Lineage-Specific Clusters within a Tetraploid Descendant

The YGOB indicates the common ancestry, prespeciation, in *Saccharomyces cerevisiae* and *Candida glabrata*, of two separate gene lineages, labeled A and B in both genomes. To apply Definition 2, we first masked the identity of all lineage B genes without deleting them from their positions, and then applied the criterion to the lineage A genes to produce the edges in G_{SCA} and G_{CGA}. We then reversed roles of A and B, masking the identity of all lineage A genes without deleting them from their positions, and then applied the criterion to the lineage B genes to obtain G_{SCB} and G_{CGB}.

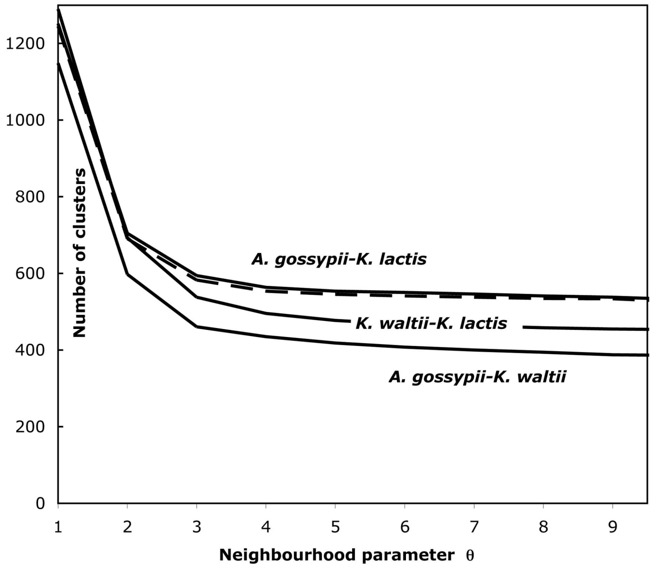


Fig. 3. Dependence of the number of clusters on neighborhood parameter, showing that, independent of θ, *K. waltii* has fewer (larger) clusters when compared with the other two genomes, as might be expected from the closer phylogenetic relationship of the latter in Fig. 2. The dashed line indicates that the max-gap criterion returns fewer, larger clusters for the same value of θ—one max-gap cluster may contain several GA clusters. Downward slope of all lines due to the incorporation of smaller clusters into larger ones as θ increases, demonstrating that almost all max-gap clusters have also conserved neighborhood structure. Max-gap clusters are constructed using the

In Fig. 4, we depict how cluster size is distributed and use this to assess the degree of relatedness of genomes or lineages. Same lineage genes across different species, SCA-CGA and SCB-CGB, form larger clusters, and thus are evolutionary closer than different lineage genes in same species, SCA-SCB and CGA-CGB.

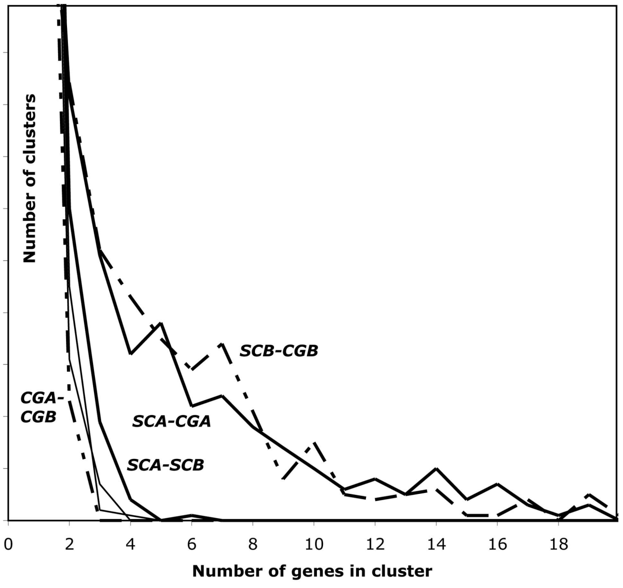


Fig. 4. Distribution of size of clusters for $\theta = 2$, showing larger clusters, i.e., less evolutionary divergence, between same lineage SCA-CGA and SCB-CGB in different species than between different lineages. Also, the two different lineages are more diverged in CG than in SC, as confirmed for larger θ (not shown), consistent with the observation of smaller duplicated blocks in *C. glabrata* than in *S. cerevisiae* from past analysis [3]. Two thinner, unlabeled curves indicate SCA-CGB and SCB-CGA.

4 Gene Clusters at the Ancestral Nodes of the Yeast Phylogeny

In this section, we transcend clusters constructed from only two genomes and proceed to reconstruct clusters at all the ancestral, i.e., nonterminal, nodes of the tree. In Section 4.1, we describe the algorithm we used, and in Sections 4.2 and 4.3, we give some results from the yeast data.

4.1 Optimizing Ancestral Nodes Minimizing Edge Appearances/Disappearances

Consider that the data at each terminal node associated with a given genome consist of a characteristic function χ on the set of pairs of vertices (genes) in the genome graph described in Definition 2 and illustrated in Fig. 1 in Section 2, where $\chi(g,h)=1$ indicates the presence of edge and $\chi(g,h)=0$ indicates its absence. For each ancestral genome, we wish to construct $\chi(g,h)$ for all g and h in that genome, so as to minimize the number of times χ changes value from one endpoint of a tree branch to the other, i.e., from one ancestral genome to its descendant genome, summed over all pairs (g,h) in both genomes, and summed over all branches in the tree. We will discuss this ancestral node optimization for unrooted binary trees, i.e., where each ancestral node has exactly three adjacent nodes, perhaps the simplest instance of dynamic programming on a tree [4, Chapter 2]. (This procedure is easily extended to nonbinary trees.)

Dynamic programming requires two passes. In the forward pass, from the terminal nodes toward the root R (chosen arbitrarily from among the ancestor nodes, without consequences for the results), the value of the variable χ (indicating the presence or absence of edge gh) may be established definitely at some ancestral nodes, while at other nodes, it is left unresolved until the second "trace-back" pass, when any multiple solutions are also identified. We call those edges that are definitely present at a node the *optimals*, while those that are potentially present during the forward pass the *near-optimals*. We need not discuss further these that are definitely excluded during the forward pass.

Note that the (arbitrary) designation of one ancestor node to be the root R determines, for each branch, which of its endpoints corresponds to the mother genome (the one proximal to the root), and which to the daughter (the one distal to the root). We order the nodes so that no node precedes any of its daughters. (This is always possible for a rooted tree.)

Suppose ancestral node N (other than the root R) has daughter nodes K and H. Because of the way we have ordered the nodes, by the time, we reach N during the forward pass, we have already decided, for each daughter, whether edge gh is an optimal or near-optimal. Then, if gh is optimal for both K and H, it is optimal for N. If it is optimal for only one of K and H, it is near-optimal for N. For the root node R, with three daughters, if gh is optimal for at least two of the three, then it is optimal for R. We need not consider near-optimals for R.

For the traceback, reversing direction in the same order, starting at R, if gh is optimal for a mother node and near-optimal for its daughter, then gh is promoted to optimal status in the daughter.

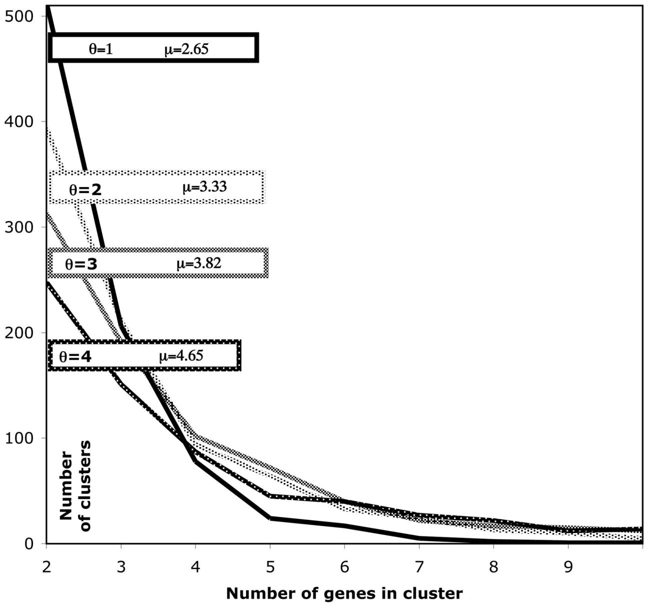


Fig. 5. Distributions of cluster size, with mean μ, at Node R, for various values of θ. Smaller clusters amalgamate into larger ones as θ increases.

Note that in this method, the presence or absence of genes in the ancestral genomes derives solely from the presence or absence of at least one edge having that gene as an endpoint.

4.2 Results: Cluster Statistics

We applied the dynamic programming method described in Section 4.1 to assign edges to each ancestor node A, B, D, Y, and R based on the five present day genomes in the yeast phylogeny. The genes in each connected component of the graph thereby constructed at an ancestral node may be considered to define a GA cluster for that node (though the details of the edge structure of this component cannot necessarily be produced as the intersection of two genome graphs). Introducing the GAs through the neighborhood parameter allows clusters to be conserved despite local rearrangements. This is seen in Fig. 5, where the distribution of cluster sizes (number of vertices) at Node R is seen to spread out to larger values as θ increases.

The average sizes of clusters are much higher in the other ancestral nodes, though they follow the same trend, as seen in Fig. 6.

While the average cluster size increases, the number of genes involved in these clusters at a given node does not change much, as seen in Fig. 7. Consequently, as seen in Fig. 7b, the number of clusters drops.

4.3 Results: Evolution and Cluster Coherency

From node to node, the number of clusters and the genes they contain change. We can, however, assess to what extent this change is gradual or abrupt. If a cluster simply gains or loses a few genes, or if a cluster divides into two, or if two clusters merge to become one, we may consider the resulting configuration a gradual change. In these cases, when we compare the outcome of such changes, the new cluster is either nested in the old one, or vice versa, or two

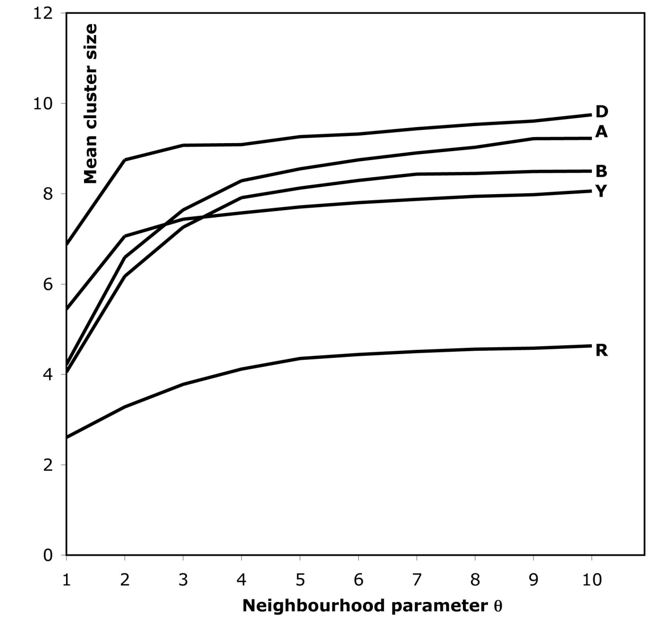


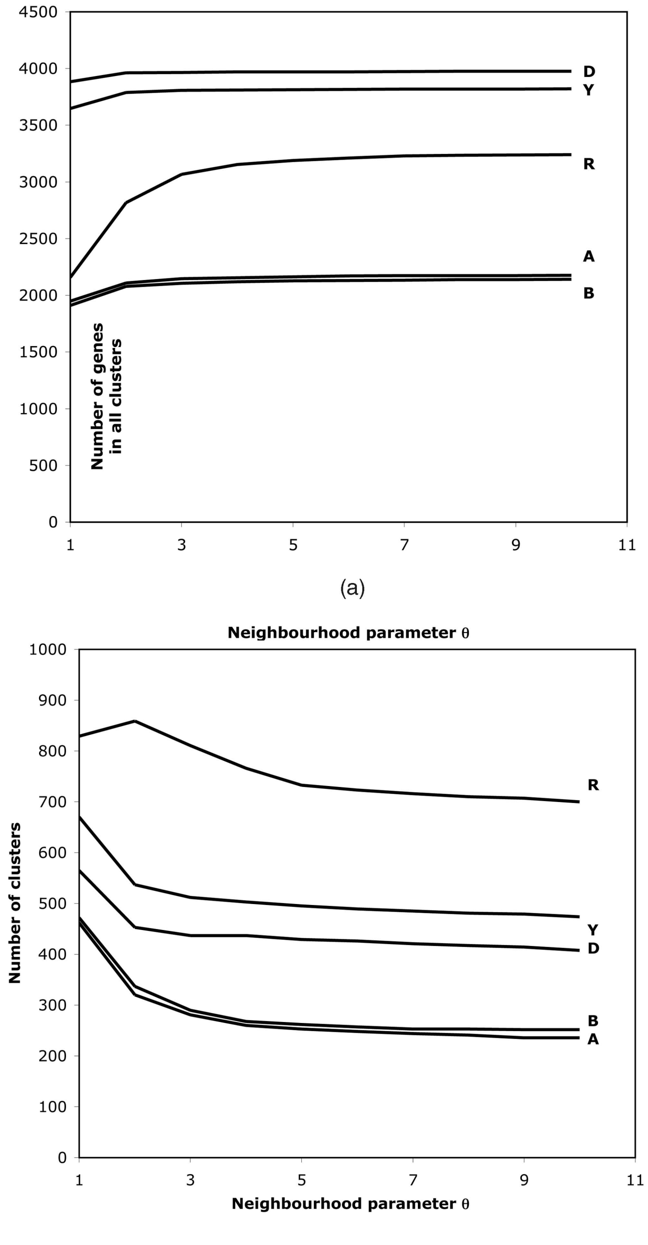
Fig. 6. Mean cluster size at ancestral nodes for various values of θ. Node R has, on average, the smallest clusters among all nodes, suggesting that it is furthest from present day genomes.

disjoint new ones correspond to an old one, or vice versa. We operationalize the notion of "gradual," then, by saying two clusters, one in each of two genomes, are in conflict unless one is nested in the other or they are disjoint. Table 1 shows what proportion of each ancestor's clusters is in conflict with their adjacent nodes' clusters according to this operationalization. Cluster evolution has been exceedingly gradual among the diploid genomes, but a good proportion of the A and B lineage clusters is seriously disrupted in their common ancestor, and vice versa.

5 Bandwidth of the Clusters

We have constructed clusters of genes based on adjacencies presumed to have been present in the ancestral genomes. While these are most parsimonious inferences, they are not sufficient to reconstruct the entire genomes, mainly because we have tried to compute neither how to partition the clusters among chromosomes nor how to impose a linear order within a cluster. Indeed, the dynamic programming is not even able to ensure that the clusters are compatible with the GA structure imposed on the data genomes in Definition 2. In other words, there is no constraint on the connected components, and hence, the entire graph inferred at an ancestral node, to have bandwidth $\leq \theta$. If the bandwidth is larger, it means that we can construct no genome where the vertices in the connected component in question can be linearly disposed, so that each edge has less than θ genes intervening between the two endpoints.

On the other hand, there is no compelling reason to insist on this bandwidth restriction on the ancestral genomes. Our initial goal was to find how clusters of vertices are preserved or evolve along various evolutionary lineages, and if the bandwidth is larger at some ancestor, this simply suggests that the cluster was looser at that time.



(a)

(b)

Fig. 7. (a) Total number of genes in clusters is remarkably stable, except for Node R, which recruits more genes up to $\theta = 4$. (b) As θ increases, small clusters are amalgamated with larger ones, so that the total number of clusters decreases.

Whatever the importance or whatever the interpretation we attach to bandwidth, it is thus of great importance to see how it is preserved or changed in the ancestral genomes we are investigating.

5.1 Algorithms and Results

The previous section describes the motivation for investigating the bandwidth of the graphs we constructed at the ancestral nodes. The problem of inferring the bandwidth of a graph is, however, NP-hard [9]. For a given θ, Saxe [10] showed that determine whether bandwidth is no greater than θ could be done in $O(n^{\theta+1})$ time and $\Theta(n^{\theta+1})$ space, where n is the number of vertices. The space was required because they employed an array to store plausible partial

TABLE 1
Conflicts in Clusters between Genomes at Two Ends of Each Tree Branch As a Function of θ

		Neighbourhood parameter		
Node	Adjacent Node	1	3	8
A	R	20	36	37
B	R	23	36	40
R	A	10	16	16
R	B	11	16	17
R	Y	0	0	0
D	Y	0	1	1
Y	D	0	0	1
Y	R	0	1	1

Percentage conflict out of the total number of clusters in genome in left-hand column.

TABLE 2
Bandwidth of Edge Sets Produced by Dynamic Programming at the Ancestral Nodes of Yeast Evolutionary Tree for Various Values of the Neighborhood Parameter θ

θ	Node				
	A	R	B	Y	D
1	1	1	1	1	1
2	2	3	2	2	2
3	3	4	3	3	3
4	4	4	4	4	4
5	5	5	5	5	5
6	6	6	6	6	6
7	7	8	7	7	7
8	8	8	8	8	8
9	9	9	9	9	9
10	10	$> 10?$	10	10	10

layouts. Gurari and Sudborough [6] simplified and improved Saxe's algorithm so that it runs in $O(n^\theta)$ time and $\Theta(n^\theta)$ space, in which an array is still used to store all plausible partial layouts. We implemented this version in a C++ program but using dynamic balanced binary search tree (i.e., `<map>` in C++) as the data structure to store the plausible partial layouts. This implementation uses $\Theta(K)$ space, where K is the actual number of partial layouts and is expected much less than $\Theta(n^\theta)$, in the expense of an extra $O(\log n)$ search time. Note that although these algorithms are also based on the principles of dynamic programming, they have no other connection with the ancestral node optimization discussed in Section 4.1, which is a completely different problem.

We ran the program at the High-Performance Computing Virtual Laboratory (HPCVL) on a Sun Fire 25,000 Node equipped with 72X dual-core UltraSPARC-IV+ 1.8-GHz processors and 576 GB of RAM. For a typical graph of ancestor R with 86 edges and 26 vertices, our program took 20 minutes to verify bandwidth ≤ 7 and occupied over 8 GB of memory while running.

For large graphs, our program can still be computationally costly, so that we use it on individual clusters only if rapid heuristics are unable to ensure that the bandwidth is no greater than θ. This happened in 162 out of the 22,780 clusters, considering θ from 1 to 10, and all five ancestral nodes. The heuristic which sufficed for the 22,618 remaining clusters was the reverse Cuthill-McKee (RCM) algorithm [5], [8]. Since the results of RCM depend on the input order of the vertices, we ran the algorithm 10,000 times with different orders to see if the estimate for the bandwidth was no larger than θ. We resort to Gurari-Sudborough algorithm when RCM cannot guarantee that bandwidth $\leq \theta$. Though the algorithms only test whether the bandwidth is less than a given value k, in each case, we used Gurari-Sudborough on selected components to prove that it was also not less than $k-1$, i.e., it was exactly the value reported in the table.

Table 2 shows the results of all our analyses, RCM followed when necessary by Gurari-Sudborough. We have no result for ancestor R when we try to verify bandwidth ≤ 10, because the program did not terminate after one week while using 300 GB of memory. As can be seen in the table, the bandwidth is never greater than θ except in a few cases for node R. In other words, the GA clusters at each ancestral node are compatible with at least one (and probably many) genome whose chromosomes are strictly linear.

These results are of interest because they show that the neighborhood parameter necessary to account for the cluster structure at an ancestral node is quite stable, i.e., generally no greater than the parameter used for the genomes at the given terminal nodes. It would be interesting to see to what extent this might break down if the phylogeny were multifurcating instead of binary branching.

6 Simulations

We undertook simulations to ascertain whether the configurations of clusters at the various ancestral nodes, for the various values of θ, could be accounted for by a simple model of random genome rearrangement along the branches of the tree in Fig. 2. The first step was to determine the number of rearrangements for each branch. Starting with the previously inferred ancestor D, but with a randomized gene order, random rearrangements were introduced in a scenario leading to a simulated KL and, independently, to a simulated AG. The number of clusters produced (for $\theta = 5$) by comparing the current stage of KL and AG was monitored until it was as close to the number previously inferred for D, based on the true KL and AG genomes. (No effort was made to optimize: the numbers of rearrangements were tested in groups of 10.) Similarly, starting with a randomized AB ancestor, simulated SC and CG genomes were generated with random rearrangements so that the clusters induced at AB by the simulated modern SC and CG were as numerous as these induced by the real genomes. Similarly, for the numbers of rearrangements needed to produce Y from D and KW, and finally to produce R from Y and AB, taking into account the doubling event at R.

Once the number of rearrangements necessary, summarized in Table 3, were estimated by these methods, the actual schema of rearrangements are applied in appropriate numbers, starting from the ancestor R as reconstructed in YGOB, and continuing through the other ancestral nodes until all five present day genomes were simulated. The random rearrangements, both here and in our above method for estimating the number for each branch, consist of inversions and translocations in a 10:1 ratio. To simulate a random inversion, two breakpoints are sampled (according to uniform probabilities) along one chromosome, and

TABLE 3
Rearrangements Applied along Branches of Phylogeny

branch	number of rearrangements
R to AB	20
R to Y	10
Y to D	20
Y to KW	70
D to AG	140
D to KL	140
AB to SC	180
AB to CG	190

the sequence of genes between the two breakpoints is reversed. To simulate a random translocation, a breakpoint is chosen on each of two chromosomes, and the prefixes or suffixes are swapped.

After the rearrangements were carried out, the same cluster construction was performed on the simulated data as on the original genomes. Fig. 8 shows the number of clusters at each ancestral node. It is not surprising that for $\theta = 5$, the simulated values and real values tend to coincide, for it was at this value of θ that we estimated the number of rearrangements to use in the simulations. However, the number of clusters in the real and simulated ancestors is also quite parallel at $\theta = 2$, 10, and 20.

These results indicate that the GA cluster approach is a robust and coherent way of approaching the cumulative perturbations in gene order due to evolutionary inversions and translocations. A simple model of inversion and translocation, the parameters of which are chosen to fit the data at one value of θ, works just as well at other values, and the number of clusters is well explained by the number of rearrangements that have intervened between two or more genomes.

7 Conclusions

The GAs we have introduced allow us to recognize clusters even though they have been perturbed by local rearrangements. That the max-gap criterion gives approximately the same number of clusters means that max-gap is too weak a criterion for these data in that it does not target order conservation in the clusters as much as GA does. So without the GA analysis, we would not know whether the max-gap clusters were ordered or not.

Our separation of the A and B lineages as separate phylogenetic lineages is validated by the higher number of within-lineage clusters than within-species clusters, with the *C. glabrata* genome appearing highly rearranged.

We have shown the interplay of bandwidth considerations and the dynamic programming optimization of ancestral nodes in a given phylogeny. Our implementation of a difficult bandwidth algorithm is a potentially useful tool.

The neighborhood parameter allows us to control the distribution of cluster sizes and the number of clusters. It allows us to explore the trade-off between the size of clusters and the rate of conflict between clusters in connected ancestral nodes.

Supplementary information. Experimental software used in this work may be obtained from the first author.

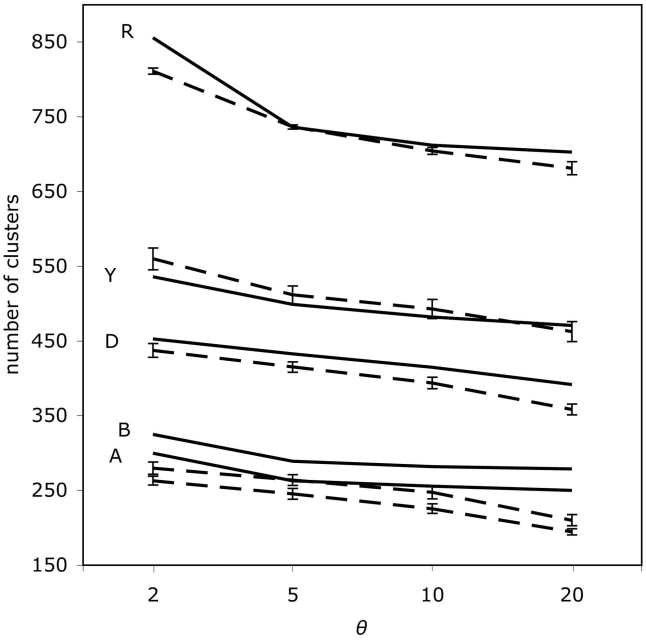


Fig. 8. Clusters reconstructed at ancestral nodes. Solid lines: Real data. Dashed lines: Average of five simulations.

Acknowledgments

The authors thank Ken Wolfe, Kevin Byrne, and Jonathan Gordon for encouragement and valuable information, including key data beyond the inestimable resource they have made available to the yeast and genomics communities. They also thank Howard Bussey for a helpful discussion and Robert Warren for his sustained collaboration in the polyploid genomics project. This research is supported in part by a grant to David Sankoff from the Natural Sciences and Engineering Research Council of Canada (NSERC). David Sankoff holds the Canada Research Chair in Mathematical Genomics.

References

[1] A. Bergeron, S. Corteel, and M. Raffinot, "The Algorithmic of Gene Teams," *Proc. Second Int'l Workshop Bioinformatics (WABI '02),* D. Gusfield and R. Guigo, eds., pp. 464-476, 2002.

[2] K.P. Byrne and K.H. Wolfe, "The Yeast Gene Order Browser: Combining Curated Homology and Syntenic Context Reveals Gene Fate in Polyploid Species," *Genome Research,* vol. 15, pp. 1456-1461, 2005.

[3] B. Dujon et al., "Genome Evolution in Yeasts," *Nature,* vol. 430, pp. 35-44, 2004.

[4] J. Felsenstein, *Inferring Phylogenies.* Sinauer Assoc., 2004.

[5] A. George, "Computer Implementation of the Finite Element Method," Technical Report STAN-CS-71-208, Computer Science Dept., Stanford Univ., 1971.

[6] E.M. Gurari and I.H. Sudborough, "Improved Dynamic Programming Algorithms for Bandwidth Minimization and the Mincut Linear Arrangement Problem," *J. Algorithms,* vol. 5, pp. 531-546, 1984.

[7] R. Hoberman, D. Sankoff, and D. Durand, "The Statistical Analysis of Spatially Clustered Genes under the Maximum Gap Criterion," *J. Computational Biology,* vol. 12, pp. 1081-1100, 2005.

[8] J. Liu and A. Sherman, "Comparative Analysis of the Cuthill-Mckee and the Reverse Cuthill-Mckee Ordering Algorithms for Sparse Matrices," *SIAM J. Numerical Analysis,* vol. 13, pp. 198-213, 1975.

[9] C.H. Papadimitriou, "The NP-Completeness of the Bandwidth Minimization Problem," *Computing,* vol. 16, pp. 263-270, 1976.

[10] J. Saxe, "Dynamic-Programming Algorithms for Recognizing Small-Band-Width Graphs in Polynomial Time," *SIAM J. Algebraic and Discrete Methods,* vol. 1, pp. 363-369, 1980.

[11] X. Xu and D. Sankoff, "Tests for Gene Clusters Satisfying the Generalized Adjacency Criterion," *Proc. Brazilian Symp. Bioinformatics (BSB '08),* A.L.C. Bazzan, M. Craven, and N. F. Martins, eds., pp. 152-160, 2008.

Qian Zhu is currently working toward the bachelor's degree specializing in biochemistry and computer science at the University of Ottawa. He has been a research intern in the Sankoff Lab during his undergraduate studies and has developed an expertise in genomic databases, notably Gramene and the Yeast Gene Order Browser.

Zaky Adam received the MSc degree in computer science from the University of Western Ontario, with advisor Lila Kari. He is currently working toward the PhD degree in computer science at the University of Ottawa and is a research assistant in the Sankoff Lab. He is interested in phylogenetic analysis, particularly various approaches to rearrangement-based phylogeny.

Vicky Choi received the PhD degree in computer science from Rutgers University. She has been an assistant professor at Virginia Tech since 2004. Her main research area is the design, implementation, and analysis of algorithms.

David Sankoff received the PhD degree in mathematics from McGill University under the direction of Donald Dawson. He has been a member of the Centre de Recherches Mathématiques in Montreal for many years. He currently holds the Canada research chair in mathematical genomics in the Mathematics and Statistics Department, University of Ottawa, and is cross-appointed to the Biology and the Computer Science Departments. His research interest is comparative genomics, particularly probability models, statistics, and algorithms for genome rearrangements.