Deriving Kripke Structures from Time Series Segmentation Results

Satish Tadepalli, Naren Ramakrishnan, Bud Mishra, Layne T. Watson, and Richard F. Helm

***Abstract*— Kripke structures are important modeling formalisms to understand the behavior of reactive systems. We present an approach to automatically infer Kripke structures from time series datasets. Our algorithm bridges the continuous world of time profiles and the discrete symbols of Kripke structures by incorporating a segmentation algorithm as an intermediate step. This approach identifies, in an unsupervised manner, the states of the Kripke structure, the transition relation, and the properties (propositions) that hold true in each state. We demonstrate experimental results of our approach to understanding the interplay between key biological processes.**

I. Introduction

Kripke structures are widely used as a modeling formalism for reasoning with temporal logics. Given a Kripke structure, algorithms for model checking in various modal (e.g., temporal) logics are a well studied topic [1]. However, the process of modeling a Kripke structure is often considered a domain-specific activity. In this paper, we present an approach to automatically reconstruct Kripke structures from time series data through the use of segmentations as an intermediate representation. We show the applicability of this technique in understanding dynamic temporal relationships between biological processes under different experimental conditions.

The input datasets we consider are multiple time series profiles (e.g., gene expression data over a time course). Each time series vector denotes an individual gene and a labeling function (over genes as well as groups of genes) is assumed to be provided. To arrive at Kripke structures from such datasets, we marry two threads of research in our work: algorithms for time series segmentation, and inferring (symbolic) temporal relationships from dynamic data.

Time series segmentation is an important data mining problem that can be used to infer the critical events occurring in a system. Segmenting a single time series has been extensively studied. Variations of dynamic programming [2], [3] and Bayesian approaches [4] have been applied here. When multiple time series are involved, it is assumed that all the series have similar patterns in a given segment. Algorithms based on fuzzy clustering [5] and graphical models [6] have been applied in this context. Essentially, all these works are based on homogeneity assumptions within segments and model the segmentation problem as one of clustering time points with the constraint that data samples in a cluster must belong to successive time points.

S. Tadepalli, N. Ramakrishnan, and L.T. Watson are with the Department of Computer Science, Virginia Tech, Blacksburg, VA 24061, USA stadepal@vt.edu

B. Mishra is with Courant Institute of Mathematical Sciences, New York University, NY 10012, USA

R. Helm is with Department of Biochemistry, Virginia Tech, Blacksburg, VA 24061, USA

Algorithms to mine the temporal order of events occurring in multiple time series have also been well studied. Moerchen et al. [7] devised a temporal grammar for this purpose. However, their approach requires manual partitioning of the time series, and the events are derived by naive discretization of the multiple time series. Kirshner et al. [8] describe a method to model multiple time series using hidden Markov models coupled with Chow-Liu trees. This model captures the mutual dependence between the multiple time series while taking into account the temporal dependencies within each individual time series.

Our work presents the first synthesis of these two distinct lines of research into an integrated approach for reconstructing Kripke structures.

II. From Time Series to Segmentations to Kripke Structures

To transduce from time series to segmentations to Kripke structures, we first model each segment of the time series as a mix of multiple clusters (of vectors). The problem of time series segmentation is then to identify the segments such that the clusters in a segment are highly dissimilar from the clusters in adjacent segments. We consider each segment as a separate dataset where the samples are multiple real valued vectors. Formally, given a two-component dataset $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, where $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$ and $\mathcal{Y} = \{\mathbf{y}_i\}_{i=1}^N$, we seek r clusters in $\mathcal{X}$ and c clusters in $\mathcal{Y}$ that satisfy two conflicting criteria. The first criterion is that the clusters are highly dissimilar across the datasets, i.e., the samples that are clustered together in $\mathcal{X}$ are clustered together with entirely different sets of samples in $\mathcal{Y}$. The second criterion is that the clusters are local in each dataset, i.e., data samples having similar profiles are grouped together in each dataset.

A. Clustering across datasets based on contingency tables

We begin by measuring the similarity of the clusters across the datasets using a $r \times c$ contingency table. Entry n_{ij} in the $(i,j)^{th}$ cell of the table represents the overlap between the samples clustered together in cluster i of $\mathcal{X}$ and in cluster j of $\mathcal{Y}$. The sizes of the clusters in $\mathcal{X}$ are given by the column-wise sums across each row: $n_{i\cdot} = \sum_j n_{ij}$, while the sizes of clusters in $\mathcal{Y}$ are given by row-wise sums down each column: $n_{\cdot j} = \sum_i n_{ij}$. Suppose we have 18 data samples and 3 clusters in each dataset, then the ideal set of highly dissimilar clusters gives rise to a contingency table as shown below:

2	2	2
2	2	2
2	2	2

Interpreting each row and column as a probability distribution, ideal highly dissimilar clusters result in a total of $(r + c)$ uniform distributions across the rows and columns of the contingency table. To capture the deviation of these distributions from the uniform distribution, we define r random variables $R_i,\ i = 1, \ldots, r$ occurring with probability $p_{R_i}(R_i = j) = \frac{n_{ij}}{n_{i.}}$ corresponding to each row. Similarly we define c random variables $C_j,\ j = 1, \ldots, c$ occurring with probability $p_{C_j}(C_j = i) = \frac{n_{ij}}{n_{.j}}$ corresponding to each column. We capture the deviation of these distributions from the uniform distributions over the rows ($U(\frac{1}{c})$) and columns ($U(\frac{1}{r})$) with

$$\mathcal{F} = \frac{1}{r}\sum_{i=1}^{r} D_{KL}(p_{R_i}||U(\frac{1}{c})) + \frac{1}{c}\sum_{j=1}^{c} D_{KL}(p_{C_j}||U(\frac{1}{r})), \tag{1}$$

where $D_{KL}(p||q) = \sum_x p(x)\log_2\frac{p(x)}{q(x)}$ is the Kullback-Leibler (KL) divergence between two probability distributions $p(x)$ and $q(x)$. We propose to cluster datasets $\mathcal{X}$ and $\mathcal{Y}$ using $\mathcal{F}$ as the objective function and minimizing it in order to yield highly dissimilar clusters across datasets $\mathcal{X}$ and $\mathcal{Y}$.

B. Clustering within datasets

In order to group data samples with similar profiles within a dataset, we define cluster prototypes $\mathbf{m}_i^{(x)},\ i = 1, \ldots, r$ for the clusters in dataset $\mathcal{X}$. The assignment of a data vector $\mathbf{x}_k$ to the clusters is given by the probability distribution $p_{\mathbf{x}_k}(V^{(\mathbf{x}_k)} = i) = v_i^{(\mathbf{x}_k)},\ i = 1, \ldots, r$, where $\sum_{i=1}^{r} v_i^{(\mathbf{x}_k)} = 1$. The probabilities $v_i^{(\mathbf{x}_k)}$ are the cluster membership indicator variables, i.e., the probability that data sample k is assigned to cluster i. Similar cluster prototypes $\mathbf{m}_j^{(y)}$, distributions $p_{\mathbf{y}_k}(V^{(\mathbf{y}_k)})$, and cluster indicator variables $v_j^{(\mathbf{y}_k)}$ are defined for the $\mathbf{y}$ vectors as well. These cluster membership probabilities are calculated as a function of the distance between the data vector and cluster prototypes in each individual dataset. The contingency table counts in the objective function $\mathcal{F}$ in Eq.(1) can be calculated as: $n_{ij} = \sum_k v_i^{(\mathbf{x}_k)} v_j^{(\mathbf{y}_k)},\ n_{i.} = \sum_k v_i^{(\mathbf{x}_k)},\ n_{.j} = \sum_k v_j^{(\mathbf{y}_k)}$. Thus we can effectively parametrize the objective function $\mathcal{F}$ in terms of the prototypes of the clusters in each individual dataset. However, assigning each data vector to the nearest cluster with probability 1 renders the function $\mathcal{F}$ non-differentiable at certain points and we cannot leverage classical numerical optimization algorithms to minimize $\mathcal{F}$. In order to avoid this problem, we parametrize the cluster prototypes using a continuously differentiable function as follows. We define

$$\gamma_{(i,i')}(\mathbf{x}_k) = \frac{||\mathbf{x}_k - \mathbf{m}_{i'}^{(x)}||^2 - ||\mathbf{x}_k - \mathbf{m}_i^{(x)}||^2}{D}, 1 \le i, i' \le r,$$

where D is the point-set diameter

$$D = \max_{k,k'} ||\mathbf{x}_k - \mathbf{x}_{k'}||^2, 1 \le k, k' \le N.$$

A well known approximation to $\min_{i'} \gamma_{(i,i')}(\mathbf{x}_k)$ is the Kreisselmeier-Steinhauser (KS) envelope function [9] given by

$$KS_i(\mathbf{x}_k) = \frac{-1}{\rho}\ln\Big[\sum_{i'=1}^{r}\exp(-\rho\gamma_{(i,i')}(\mathbf{x}_k))\Big],$$

where $\rho \gg 0$. The KS function is a smooth function that is infinitely differentiable. Using this the cluster membership indicators are redefined as:

$$v_i^{(\mathbf{x}_k)} = Z(\mathbf{x})^{-1}\exp\Big[\rho KS_i(\mathbf{x}_k)\Big], \tag{2}$$

where $Z(\mathbf{x})$ is a normalizing function such that $\sum_{i=1}^{r} v_i^{(\mathbf{x}_k)} = 1$. The cluster memberships for the dataset $\mathcal{Y}$, $v_j^{(\mathbf{y}_k)}$, are also smoothed similarly,

$$v_j^{(\mathbf{y}_k)} = Z(\mathbf{y})^{-1}\exp\Big[\rho KS_j(\mathbf{y}_k)\Big]. \tag{3}$$

This formulation results in soft cluster membership probabilities, where a data sample is assigned to the nearest cluster with probability slightly less than 1 and all the other clusters with a probability slightly greater than 0.

Using Eq.(2) and Eq.(3) to calculate contingency table counts and minimizing $\mathcal{F}$ should ideally lead to clusters that are local in each dataset and maximally dissimilar across datasets. However, there is a potential degenerate solution where each data vector is assigned with equal probability to all the clusters. In the example with 18 samples described earlier, each data sample can be assigned with probability $[1/3, 1/3, 1/3]$ and the resultant contingency table counts would still be uniform in each cell ($\sum_k v_i^{(\mathbf{x}_k)} v_j^{(\mathbf{y}_k)} = 2$). To avoid degenerate solutions such as these, we maximize the deviation between the cluster assignment probability of each individual data vector and the uniform distribution over the number of clusters. This results in a regularized objective function

$$\begin{aligned}\mathcal{F} =& \frac{\lambda}{r}\sum_{i=1}^{r} D_{KL}(p_{R_i}||U(\frac{1}{c})) + \frac{\lambda}{c}\sum_{j=1}^{c} D_{KL}(p_{C_j}||U(\frac{1}{r})) \\ &- \frac{1}{N}\sum_{k=1}^{N} D_{KL}(p_{\mathbf{x}_k})||U(\frac{1}{r})) \\ &- \frac{1}{N}\sum_{k=1}^{N} D_{KL}(p_{\mathbf{y}_k})||U(\frac{1}{c})).\end{aligned} \tag{4}$$

C. Segmentation algorithm

Given multiple vectors of measurements $\mathcal{G} = \{\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_N\}$, where each $\mathbf{g}_i$ is a time series over $\mathcal{T} = \{t_1, t_2, \ldots, t_l\}$, the problem of segmentation is to find a sequence of segments, $(w_{t_1}^{t_a}, w_{t_{a+1}}^{t_b}, \ldots, w_{t_k}^{t_l})$ where each segment $w_{t_s}^{t_e}$, $t_s \le t_e$, is a set of consecutive time points beginning at time point t_s and ending at time point t_e. The clusters in the adjacent segments satisfy the two conflicting criteria as described earlier. We use a window based approach to find the sequence of segments with maximally dissimilar clusters as shown in Fig. 1. Optimization of

input $\mathcal{T} = (t_1, t_2, \ldots, t_l)$: Given time series data sequence.
input l_{min}: Minimum window length.
input l_{max}: Maximum window length.

Step 1: Define the set of windows starting from time point t_a, $S_{t_a} = \{w_{t_a}^{t_b} | l_{min} \leq t_b - t_a + 1 \leq l_{max}\}$.
Step 2: Construct a directed acyclic graph where each $w_{t_a}^{t_b}$ is a node and a directed edge exists from $w_{t_a}^{t_b}$ to the windows $w_{t_{b+1}}^{t_c} \in S_{t_{b+1}}$.
Step 3: Cluster the adjacent windows $w_{t_a}^{t_b}$ and $w_{t_{b+1}}^{t_c}$ by minimizing the objective function in Eq.(4). Let $\mathcal{F}_{w_{t_a}^{t_b}, w_{t_{b+1}}^{t_c}}$ be the final value of the objective function. Assign $\mathcal{F}_{w_{t_a}^{t_b}, w_{t_{b+1}}^{t_c}}$ as the edge weight between the nodes represented by $w_{t_a}^{t_b}$ and $w_{t_{b+1}}^{t_c}$.
Step 4: Let $E_{t_l} = \{w_{t_k}^{t_l} | l_{min} \leq t_l - t_k + 1 \leq l_{max}\}$ be the set of windows ending in the last time point t_l. For each window starting at the first time point, $w_{t_1}^{t_b} \in S_{t_1}$, calculate the minimum cost path to all $w_{t_k}^{t_l} \in E_{t_l}$.
Step 5: Calculate $D_{Seg} = \mathcal{F}_{\{w_{t_1}^{t_a}, w_{t_{a+1}}^{t_b}\}} + \mathcal{F}_{\{w_{t_{b+1}}^{t_c}, w_{t_{c+1}}^{t_d}\}} + \ldots + \mathcal{F}_{\{w_{t_j}^{t_k}, w_{t_{k+1}}^{t_l}\}}$ for each shortest path in step 4.
Step 6: Return the path with minimum D_{Seg} .

Fig. 1. Algorithm for segmenting a time series

$\mathcal{F}$ for clustering adjacent segments is performed using the augmented Lagrangian algorithm with simple bound constraints on the cluster prototypes using the FORTRAN package LANCELOT [10].

D. Inferring Kripke Models from Segmentations

Gantt charts are an effective way to visualize segmentations of one time series. Kripke structures help understand the interplay between multiple segmentations. We assume the availability of a labeling function that assigns state symbols to clusters in a given segment. Each cluster can be labeled with one or more symbols and thus each segment can have multiple symbols. If K_{S_1} is the set of states active in a particular segment S_1 and K_{S_2} is the set of states active in the immediately following segment S_2, we assume that all states in K_{S_2} are reachable from all the states in K_{S_1}. These transitions can be projected down to a smaller set of symbols, for comprehensibility purposes.

The labeling function can be defined in many ways. Here, we perform an enrichment analysis of the clusters by calculating the hypergeometric probability of the genes in the cluster with respect to ontologies such as the GO (Gene Ontology) biological process (BIO) taxonomy. We choose GO BIO categories with a p-value $< 10^{-7}$ as the states active in a segment.

In the next section, we present specific examples of Kripke models inferred by segmenting gene expression datasets.

III. Inferring Kripke models from gene expression data through time series segmentation

A. Datasets

We present the time series segmentation results from two gene expression data sets in this section. The first data set is the yeast metabolic cycle (YMC) from Tu et al. [11]. The yeast metabolic cycle consists of a reductive charging (**R/C**) phase involving non-respiratory metabolism (glycolysis, fatty acid oxidation) and protein degradation, followed by oxidative metabolism (**Ox**), where respiratory processes are used to generate adenosine triphosphate (ATP), culminating in reductive metabolism (**R/B**), characterized by a decrease in oxygen uptake and emphasis on DNA replication, mitochondrial biogenesis, and cell division. The original dataset consists of 6555 unique genes from the *S. cerevisiae* genome with gene expression measurements over 36 time points spanning 3 continuous metabolic cycles.

The second dataset is taken from the experiments conducted by Shapira et al. [12]. We use the data obtained by treating *S. cerevisiae* with hydrogen peroxide (HP) after release from a G1 arrest. Due to the oxidative stress induced by HP, the cells are later arrested in the G2/M phase without progressing through the cell cycle. This dataset consists of 6076 unique genes with gene expression measurements over 20 time points.

From both datasets, we eliminated the genes that do not have an annotation in any GO biological process category (revision 4.205 of GO released on 14 March 2007). This resulted in 3602 genes in YMC and 2471 genes in HP. The gene expression values were log transformed (base 10) and normalized such that the mean expression of each gene across all time points is zero.

B. Dataset segmentation and Kripke model state inference

We used the algorithm described in Fig. 1 to segment the datasets. Recall that our hypothesis for segmenting time series is that the clusters across segment boundaries are maximally dissimilar. We show the contingency tables for the clusters in the first cycle of YMC in Fig. 2. The segments identify the R/C, Ox, and R/B phases in order. The first row shows that the genes corresponding to the particular phase come together during the segment. The second row shows the contingency tables for cluster movement across the segment boundaries and these tables are close to a uniform distribution. These results are in accordance with our hypothesis that maximally dissimilar clusters identify segment boundaries. We define the biological processes active in a segment as the states of the Kripke model. In order to identify these biological processes, the first step is to identify the sets of genes that are significantly clustered together in a segment. Each segment except the first and last segments has two sets of clusters, one set dissimilar to the clusters in previous window and the other set dissimilar to the clusters in the next window. We are interested in the genes that are significantly clustered together in these two sets of clusters, as they

segment[1-6]

	R/C	Ox	R/B	others
W1-C1	920	102	72	100
W1-C2	80	427	377	309
W1-C3	102	374	403	336

segment[7-10]

	R/C	Ox	R/B	others
W2-C1	421	60	370	328
W2-C2	200	795	150	106
W2-C3	481	48	332	311

segment[11-14]

	R/C	Ox	R/B	others
W3-C1	509	337	34	285
W3-C2	422	458	20	280
W3-C3	191	108	778	180

	W2-C1	W2-C2	W2-C3
W1-C1	421	401	372
W1-C2	380	448	365
W1-C3	378	402	435

	W3-C1	W3-C2	W3-C3
W2-C1	380	355	428
W2-C2	415	435	410
W2-C3	368	390	421

Fig. 2. Contingency tables from obtained from the clusters in the first cycle of YMC.

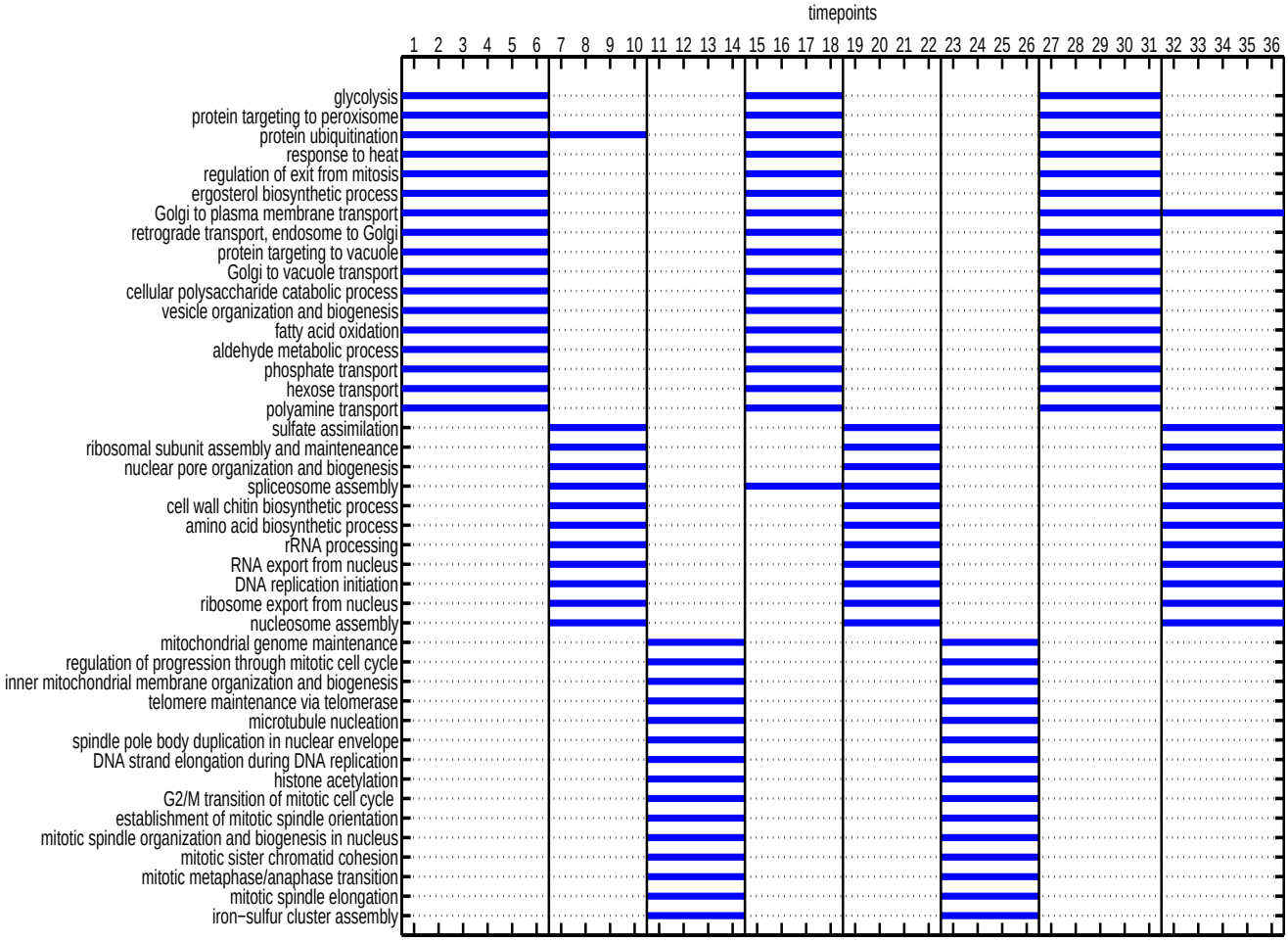

Fig. 3. Gantt chart resulting from segmentation of YMC.

represent the genes that are specific to the segment under consideration. We calculate a contingency table of these two clusterings for each segment (excluding the first and the last segment). Each cell in the contingency table represents the number of genes that are together across the sets of clusters with respect to the previous and next segments. We use a bootstrapping procedure to evaluate the significance of these sets. We randomly sample 1000 sets of clusters that are of the same size used to generate the original contingency table, and calculate the random contingency tables and this gives a random distribution for each cell of the contingency table. We now evaluate each cell of the actual contingency table with respect to the corresponding random distribution and retain only those cells that have more genes than that observed at random with $p < 0.05$ (in practice this p-value is Bonferroni corrected with the number of cross clusters to account for multiple hypothesis testing). We then perform functional enrichment over the selected sets of genes. A hypergeometric $p-$value is calculated for each GO biological process term, and an appropriate cutoff is chosen using false discovery rate (FDR) $q-$ level of 0.01 [13].

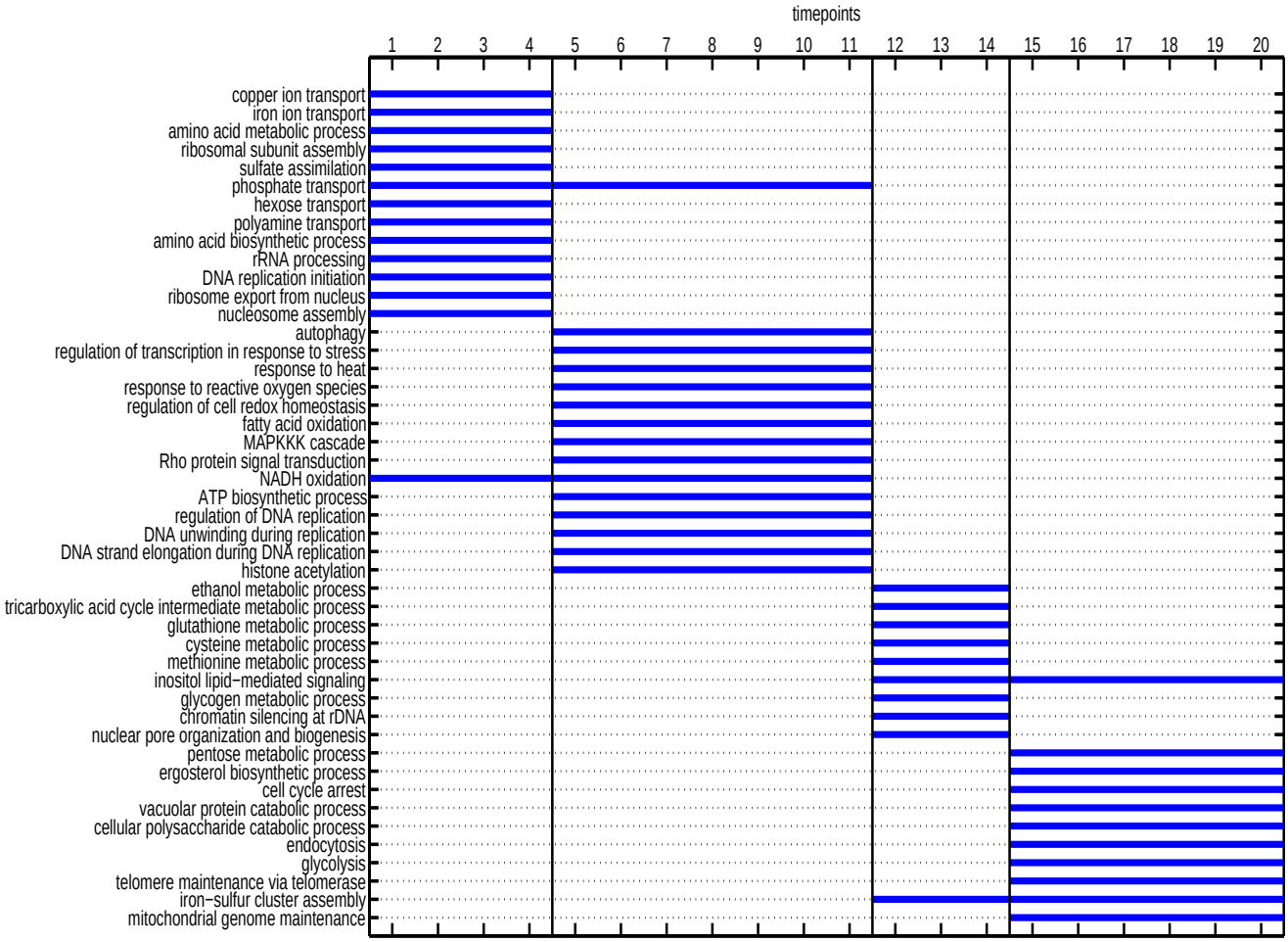

Fig. 4. Gantt chart resulting from segmentation of HP.

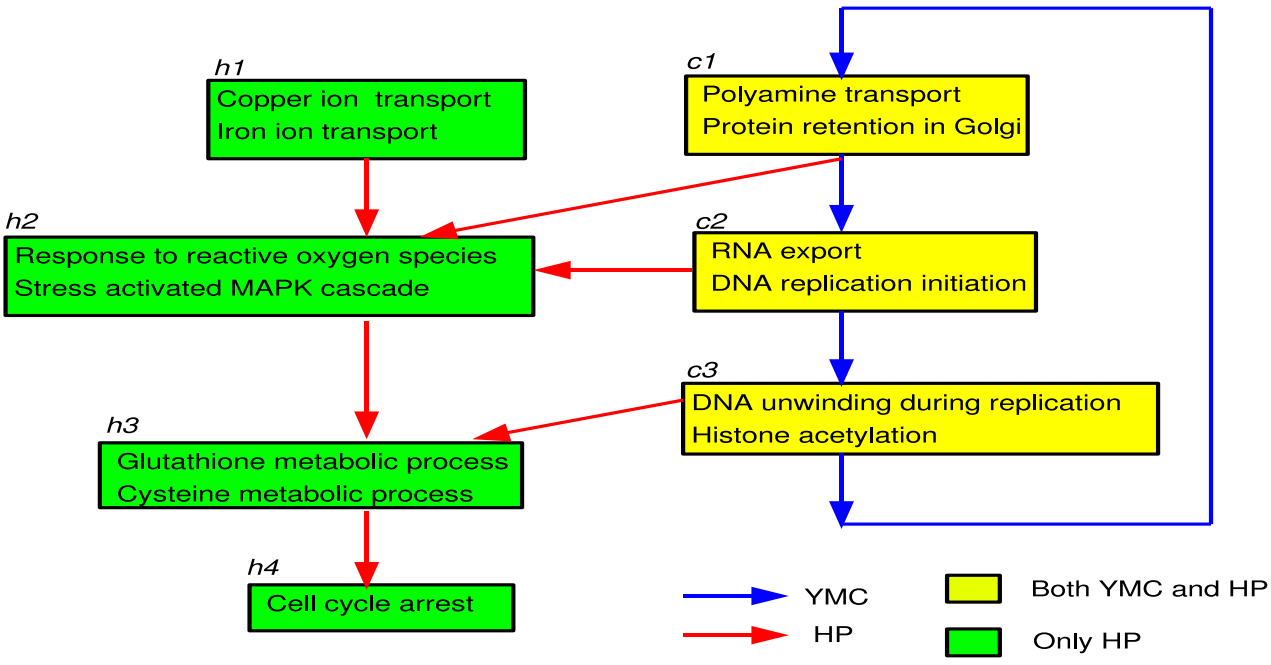

Fig. 5. Combined Kripke model.

C. Gantt chart view of the Kripke models

Figures 3 and 4 show the biological processes inferred from the datasets YMC and HP respectively with $p < 10^{-7}$. For both datasets we use a minimum window length of 3 and a maximum window length of 7. The segmentation generated for YMC is: 1-6, 7-10, 11-14, 15-18, 19-22, 23-26, 27-31, 32-36, which correspond to alternating R/C, Ox, and R/B phases. The segmentation obtained for HP is 1-4, 5-11, 12-14, 15-20, corresponding to G1, S, G2, G2/M phases of the cell cycle as depicted in Fig. 4. The cells here are arrested in the G2/M phase without cyclic progression. The biological processes enriched in each segment represent the states of the Kripke model and each of these states is reachable from the states in the immediately preceding segment. The biological processes inferred by our segmentation are in accord with results reported in the corresponding research papers.

Fig. 5 shows a combined Kripke model of the two datasets for few selected biological processes. In order to compute the combined Kripke model, we assume that each of the m processes in segment S_1 of each data set precede each of the n processes in the next segment S_2, i.e, we have an arrow representing a state transition from each process in S_1 to each process in S_2. We then find the maximal sets of processes that are common across the two datasets with the same precedence relationships. The remaining processes are specific to each individual dataset and they are also combined into bigger sets based on precedence relationships.

The yellow colored boxes in Fig. 5 represent some of the maximal sets of processes common to both YMC and HP. The green colored boxes represent the biological processes occurring only in HP. The red colored arrows show the sequence in which the processes occur in HP while the blue colored arrows represent the sequence of processes in YMC. Note that the processes end up in a cell cycle arrest in the case of HP while in the case of YMC they continue in a cyclic fashion. The processes in boxes tagged *h1, c1, c2* all occur together before the processes in the box tagged *h2* as indicated by the red arrows from *h1, c1, c2* to *h2*. The processes described in the green boxes represent the specific response of yeast to the oxidative stress induced by hydrogen peroxide. Note that these processes include MAPK cascade and glutathione metabolic process which eventually lead to cell cycle arrest as indicated by Shapira *et al.* [12].

IV. Discussion

Over the past decade, many powerful data mining techniques have been developed to analyze temporal and sequential datasets [14]. However, a formal link from such methods to an underlying temporal model has been missing. This is precisely the void that we have sought to fill in this paper. Our approach of using segmentations as an intermediate representation helps capture the dynamics of important processes from temporal datasets. Besides the applications described here, our work will have important uses in manufacturing systems, automotive engineering, and physical plant processes, all of which pose problems of network discovery from large, temporal, data streams. Our emphasis on integrating formal models with data mining allows us to leverage the large body of literature in model checking and simulation to extract not just patterns from data but complete, executable, computational models [15].

References

[1] E. Clarke, O. Grumberg, and D. A. Peled, Model Checking. The MIT Press, 2000.

[2] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "Segmenting time series: A survey and novel approach," in Data Mining in Time Series Databases. World Scientific Publishing Company, 2003.

[3] J. Himberg, K. Korpiaho, H. Mannila, J. Tikanmaki, and H. Toivonen, "Time series segmentation for context recognition in mobile devices," in ICDM, 2001, pp. 203–210.

[4] P. Fearnhead, "Exact and efficient bayesian inference for multiple changepoint problems," Statistics and Computing, vol. 16, no. 2, pp. 203–213, 2006.

[5] J. Abonyi, B. Feil, S. Nemeth, and P. Arva, "Fuzzy clustering based segmentation of time-series," in Advances in Intelligent Data Analysis. Springer, 2003.

[6] X. Xuan and K. Murphy, "Modeling changing dependency structure in multivariate time series," in ICML, 2007, pp. 1055–1062.

[7] F. Mörchen and A. Ultsch, "Discovering temporal knowledge in multivariate time series," in GfKI, 2005, pp. 272–279.

[8] S. Kirshner, P. Smyth, and A. Robertson, "Conditional chow-liu tree structures for modeling discrete-valued vector time series," in UAI, 2004, pp. 317–324.

[9] G. Wrenn, "An Indirect Method for Numerical Optimization using the Kreisselmeier-Steinhauser Function," NASA Contractor Report 4220, March 1989.

[10] A. Conn, N. Gould, and P. Toint, LANCELOT: A Fortran Package for Large-scale Nonlinear Optimization (Release A). Springer Verlag, 1992, vol. 17.

[11] B. Tu, A. Kudlicki, M. Rowicka, and S. McKnight, "Logic of the Yeast Metabolic Cycle: Temporal Compartmentalization of Cellular Processes," Science, vol. 310, no. 5751, pp. 1152–1158, 2005.

[12] M. Shapira, E. Segal, and D. Botstein, "Disruption of Yeast Forkhead-associated Cell Cycle Transcription by Oxidative Stress," Molecular Biology of the Cell, vol. 15, no. 12, pp. 5659–5669, 2004.

[13] J. Storey and R. Tibshirani, "Statistical significance for genomewide studies," PNAS, vol. 100, no. 16, pp. 9440–9445, 2003.

[14] K. Casey, "Information based biclustering for the analysis of multivariate time series data," Master's thesis, Courant Institute of Mathematical Sciences, New York University, 2007.

[15] J. Fisher and T. A. Henzinger, "Executable cell biology," Nature Biotechnology, vol. 25, pp. 1239–1249, 2007.