The unrooted set covering connected subgraph problem differentiating between HIV envelope sequences

Stephen J Maher[1,2] and John M Murray[2]

[1]Zuse Institute Berlin, Takustr. 7, 14195 Berlin, Germany.

[2]School of Mathematics and Statistics, University of New South Wales, Sydney NSW 2052, Australia.

Abstract

This paper presents a novel application of operations research techniques to the analysis of HIV *env* gene sequences, aiming to identify key features that are possible vaccine targets. These targets are identified as being critical to the transmission of HIV by being present in early transmitted (founder) sequences and absent in later chronic sequences. Identifying the key features of *env* involves two steps: first, calculating the covariance of amino acid combinations and positions to form a network of related and compensatory mutations; and second, developing an integer program to identify the smallest completely connected subgraph of the constructed covariance network. The integer program developed for this analysis, labelled the unrooted set covering connected subgraph problem (USCCSP), integrates a set covering problem and connectivity evaluation, the latter formulated as a network flow problem. The resulting integer program is very large and complex, requiring the use of Benders' decomposition to develop an efficient solution approach. The results will demonstrate the necessity of applying acceleration techniques to the Benders' decomposition solution approach and the effectiveness of these techniques for solving the USCCSP.

Key words: OR in medicine, HIV *env* sequence, Benders' decomposition, acceleration techniques

Introduction

Human immunodeficiency virus (HIV) currently infects approximately 40 million people worldwide and has resulted in over 20 million deaths. Although antiretroviral therapy has reduced mortality and morbidity from this disease in developed countries, with expanding but limited

availability of therapy in developing countries, the large number of new infections each year demonstrates a need for more effective prevention programs. The most effective prevention will be provided by a vaccine, but to date no successful HIV vaccine has been developed despite a number of trials of potential candidates. One of the difficulties of developing a vaccine is the high rate of mutation of HIV that results in shifting targets for the immune response. This is particularly evident in the envelope gene *env* that codes for the gp160 protein, which is cleaved into the gp120 and gp41 glycoproteins. gp120 buds from the virion surface and is responsible for binding to the CD4 receptor of immune cells, while gp41 is responsible for fusion with the target cell membrane and mediates the resulting infection of the cell.

The most exposed regions of gp120 are susceptible to antibody binding and clearance and hence can be highly variable. An effective vaccine must stimulate the expansion of antibodies to regions of gp120 that are necessary for viable virus and are also amenable to antibody contact. The evolving nature of gp120 and gp41 over the course of infection is evidence that antibodies against HIV envelope are generated. However, their development is too slow to inhibit infection of an individual or to eliminate infection once established. A vaccine needs to prime an antibody response to very early stages of HIV. The targets for such a vaccine are currently unclear, but may be elucidated by studying how *env* changes from the early (founder) stage of infection to the chronic stage [8]. Features that are present in founder viruses but absent in chronic viruses may indicate aspects that are under eventual immune pressure, and are candidates for vaccine targets.

Initial investigations to identify amino acid features of envelope between the two groups followed the approach of Murray *et al.* [15]. In [15], the developed approach looked at amino acid pairs in the hepatitis C virus (HCV) envelope that distinguished responders to antiviral therapy. The investigation in this paper, analyses 266 HIV envelope protein sequences, 133 founder and 133 chronic, obtained from a previous study comparing glycosylation sites between founder and chronic individuals [13]. Since a total of 266 envelope sequences in the current investigation, and in other studies, is small in comparison to the 858 amino acids (AA) in HIV envelope, there can be many positions in the sequences that will express different AA for founders compared to chronics. Hence the underlying state space employed is not the individual AA, but rather covarying pairs of AA that achieves some minimal level of covariance. The reasoning being that pairs of AA would covary if there was some functional relationship between these positions in the sequence. The integer programming formulation and approach presented in [15] is applied

by Murray *et al.* [14] to identify the fewest AA pairs over the space of all covarying pairs from *env* that express particular amino acid combinations present in one group but not the other.

The integer programming formulation and the resulting analysis of Murray *et al.* [14] identifies a set of separating pairs that are usually not connected. While these pairs identify important positions in *env*, the lack of connectivity reduces how useful the positions are in the development of a vaccine against HIV. In this paper, we modify the approach of [14] to identify a set of covarying pairs that form a connected subgraph. This connected subgraph is desirable for several reasons. First, a number of compensatory mutations will be needed for envelope to sufficiently change its structure to evade the immune response. These mutations will likely result in covariance between the related positions. Second, an antibody stimulated by a vaccine will bind between 5 and 8 AA of its antigen. Therefore, relevant antibodies to early infection will result in changes to this subnetwork of AA. Finally, a network of target AA may better describe underlying mechanisms by which HIV evades immune system clearance.

In this paper an integer programming model is developed that selects a set of covarying pairs with the following properties. First, the *most important* features of the gene sequence are identified by this set. This is defined as the fewest number of pairs that express AA combinations that exist in some founder sequences but in no chronic sequences. Second, the selected covarying pairs and amino acid positions form a completely connected subnetwork of the original covariance network. The development of such an integer program presents a number of challenges in regards to the modelling and solution approaches. The contributions of this paper arise in both of these areas. The integer programming problem developed in this paper is described as the unrooted set covering connected subgraph problem (USCCSP).

1 Literature Review

The connected subgraph problem describes a broad class of problems to which the USCCSP belongs. This problem class has applications to a variety of different research fields, in particular wildlife conservation [6, 7] and network design [2]. This paper represents the first application of this problem class to the analysis of gene sequences. The fundamental aspect observed in applications of this problem class is the distribution of key features, either essential habitat regions or *important* covarying AA pairs, across a large network. The wildlife conservation applications attempt to identify contiguous regions that improve the mobility of threatened

species between reserves [6, 7] or connect known habitats of a number of different species [16–18]. Identifying a connected subnetwork that describes these important features is extremely important to minimise the cost of conservation or, in our case, identify features to aid vaccine development.

As stated previously, the USCCSP is a specific variant of the connected subgraph problem. This variant is distinguished by having no root or terminal nodes specified, implying that the connected subgraph may be found in any region of the underlying network. To the best of the authors knowledge, such a problem formulation has not been explicitly considered in the literature.

A typical example of the connected subgraph problem with multiple fixed terminal nodes is presented by Conrad *et al.* [5]. The specification of terminal nodes alters the problem formulation by providing a fixed set of vertices that must be included in the resulting subgraph. The solution approach employed by Conrad *et al.* [5] involves two key stages, i) identifying the nodes and edges to include in the subgraph and ii) checking the graph for connectivity. The latter of these stages is formulated as a network flow problem with a single source and multiple terminal locations. This work is extended by Gomes *et al.* [9], enhancing the solution approach with the introduction of a two-phase algorithm. The first phase of the algorithm presented in [9] solves a minimum Steiner tree problem to identify a feasible, but sub-optimal, connected subgraph solution. Given a feasible solution, the second phase then solves a mixed integer program to improve the solution quality.

The conservation reserve network problem considered by Önal and Briers [16, 17] and Önal and Wang [18] is similar to the problem presented in this paper. In particular, the selection of habitat regions is formulated as a set covering problem, without the specification of root or terminal nodes. However, the data used by Önal and Briers [16, 17] forces the inclusion of specific habitat sites. This requirement implicitly defines a set of root nodes for the resulting connected subgraph. A limitation of the work presented by Önal and Briers [16, 17] and Önal and Wang [18] is the use of a network based upon two dimensional data that is partitioned into a regular square grid. Consequently, there is an upper bound of 8 on the connectivity of each node, which is significantly smaller than the connectivity observed in the underlying graphs considered in this paper. A final limitation of these approaches [16–18], is the requirement that the resulting subgraph forms a spanning tree of the selected sites, preventing the possibility of cycles appearing in the optimal solution.

Connectivity in subgraphs has been evaluated using a variety of different modelling approaches. However, the permissible methods for connectivity evaluation are dependant on whether the subgraph is formed with the selection of nodes or edges. Forming subgraphs by selecting nodes is the most common approach employed, which permits the use of trees to construct connected subgraphs. The property that a tree is a completely connected graph is exploited by Önal and Briers [16,17] and Önal and Wang [18]. In addition, the Steiner tree problem is a very useful and closely related problem, which is employed by Dilkina and Gomes [7]. Alternatively, Gomes *et al.* [9], Conrad *et al.* [6] and Dilkina and Gomes [7] consider the single and multi-commodity flow problems as a method to impose connectivity constraints. Finally, the properties of node-cut sets are employed by Carvajal *et al.* [4] to impose connectivity constraints between two non-adjacent nodes that are selected in the subgraph. The connectivity evaluation approaches presented by Gomes *et al.* [9], Conrad *et al.* [6], Dilkina and Gomes [7] and Carvajal *et al.* [4] are also permissible for problems forming subgraphs through the selection of edges.

The contributions of this paper are twofold, the analysis of HIV *env* sequences and the development of the unrooted set covering connected subgraph problem. First, to the best of the authors' knowledge, apart from alignment methods, the only applications of operations research to the analysis of gene sequences are presented here and in Murray *et al.* [15] and Murray *et al.* [14]. This paper contributes to this application area with the development of a sophisticated integer programming problem to analyse HIV *env* in the pursuit of vaccine development. Second, this paper extends the connected subgraph problem class by presenting a general formulation of the unrooted set covering variant. The solution approaches presented in this paper have not been previously considered, and the novel implementation of the Benders' decomposition is a contribution of this paper.

The exposition in this paper is presented with the following structure. Section 2 provides a description of the problem and presents key details related to the application. Section 3 describes the mathematical model for the general form on the USCCSP and is used to solve the problem presented in Section 2. The solution approach employed to solve the connected subgraph problem is described in Section 4, including a variety of acceleration techniques for Benders' decomposition. An extension of the USCCSP is discussed in Section 5, detailing the modifications to the problem formulation presented in Section 3. The computational results for the original problem formulation and the extension are presented in Section 6. Finally, the

conclusions and possible future work are detailed in Section 7.

2 Problem description

The objective of the USCCSP is to select the smallest set of covarying AA pairs forming a completely connected network that express combinations observed by the founder viruses but not the chronic. To achieve this, a covariance network is constructed for each clade using all of the collected sequences, from both founder and chronic viruses. The method employed for constructing the covariance network is given by Aurora *et al.* [3]. The approach of [3] is basically a variant of a chi-squared test to determine whether the AA combinations observed at a pair of positions across all sequences differ from what would be observed from random combinations. Covarying pairs are ones that achieve a specified cut-off value. The AA combinations on each covarying pair that are displayed by any chronic virus are discarded, resulting in a network of envelope positions and their AA combinations exhibited by only founder sequences. This network is described as the founder sequence separating pairs network, hereafter the *separating pairs network*. Given the inherent variability of *env* and the large number of covarying pairs, there will be many subgraphs of this network where each founder sequence will exhibit at least one of the AA combinations contained within a separating pair of the subgraph. The *separating pairs problem* [14, 15], determines the minimal such network, which will not necessarily be connected.

The work in this paper extends [14, 15] by introducing constraints that enforce the connectivity between the selected pairs. These additional constraints ensure that a unit of flow can pass from each node to every other node in the selected subnetwork. If this is not possible, a penalty is applied for every non-selected edge that is required to permit this flow. As a result,

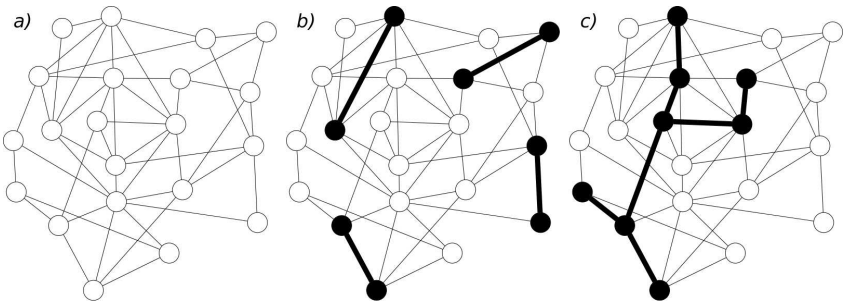


Figure 1: a) A separating pairs network. b) Solution of the separating pair problem [14, 15] (Selected edges are thick lines). c) Solution of the USCCSP.

the objective of this problem can be separated into two parts, i) minimising the number of separating pairs and ii) minimising the number of additional pairs required to form a completely connected network.

Examples of a separating pairs network, the solution to the separating pairs problem of [14, 15] and the expected result of the USCCSP are given in Figure 1. As suggested by Figure 1b, the results of [14, 15] identify a small set of pairs located throughout the separating pairs network. Imposing connectivity on the selected separating pairs potentially results in the selection of an alternative set of pairs, as observed by comparing Figures 1b and c. This can arise as a result of the distance between the covarying pairs selected by the separating pairs problem being too great to simply add the linking edges. As such, the solution to the USCCSP may result in many different separating pairs compared to the results presented in [14]. However, the connectivity requirement potentially identifies the multiple mutation steps of HIV evading a developing immune response at acute infection.

3 The unrooted set covering connected subgraph problem

The separation of the founder and chronic sequences, i.e. identifying features specific to the founder sequences but not the chronic, is achieved by selecting a set of edges such that each founder sequence exhibits at least one AA combination given by the separating pairs. To define this more formally, we introduce the set N to contain all amino acid positions j that are observed on a covarying pair. The edges of the separating pairs network are denoted by (i, j), where $i, j \in N$, and are contained in the set C. It is assumed that only one edge exists between two amino acid positions. As such, each edge (i, j) can represent multiple amino acid combinations that are exhibited by the founder but not the chronic sequences for that pair of positions, as discussed in Section 2. To formulate the separation problem, we define Q to contain all founder sequences q and the parameters a_{ijq} equal 1 if an AA combination on covarying pair (i, j) is exhibited by sequence q, 0 otherwise. Recall that founder AA combinations are excluded if they also appear on chronic sequences. Finally, the variables x_{ij} equal 1 if covarying pair (i, j) is selected, 0 otherwise, at a cost of c_{ij}.

The parameters and variables described above are used to formulate the separating pairs problem presented by Murray *et al.* [14, 15]. The extension of [14, 15] given by the USCCSP introduces connectivity requirements between each of the pairs selected by the separating integer

program. The connectivity evaluation and selection of pairs is performed simultaneously, as such the AA positions $j \in N$ in the selected pairs are unknown *a priori*. As a result, all pairs of nodes in N are used to form a set of source-sink pairs. To properly model the connectivity requirements, the set S is defined to contain all source-sink pairs $(s,t), s \in N, t \in T^s$, where $T^s = \{t \in N | t > s\}$. A network flow problem is then constructed for each $(s,t) \in S$ to identify a path from s to t.

The formulation of the network flow problem involves defining an additional set of variables for each source-sink pair contained in S. Flow variables for this formulation are given by y_{ij}^{st} that equal 1 if a selected pair (i,j) is used in the path between source-sink pair (s,t), and 0 otherwise. Since there is no guarantee that the optimal solution to the USCCSP is a completely connected network, a penalty is applied for each non-selected pair required for a unit of flow to pass from s to t. This penalty is applied with the introduction of variables z_{ij}^{st}. If $y_{ij}^{st} = 1$ and $x_{ij} = 0$, the value of z_{ij}^{st} is 1, otherwise $z_{ij}^{st} = 0$. This condition is set by the problem constraints. An additional multiplicative parameter M is applied to the z_{ij}^{st} variables in the objective function to alter the potency of the connectivity evaluation in the USCCSP.

While a network flow problem is formulated for each source-sink pair, it is only necessary to identify a path between nodes s and t if these nodes exist on at least one selected pair. This is explained with reference to a small network presented in Figure 2. In this example two edges are selected (highlighted in bold), resulting in a subnetwork containing four nodes (filled circles). Given this solution to the USCCSP, a unit of flow must be passed between each of the four selected nodes. In the model constraints, if $\max_{i,j \in N}\{x_{si} + x_{jt} - 1\} = 1$ then flow must pass between source-sink pair (s,t). If an (s,t) pair do not satisfy this constraint, because at least one of s or t does not lie on the selected pairs, then the problem minimisation will lead to a zero flow between these nodes. As a result, the cost of passing a unit of flow between the

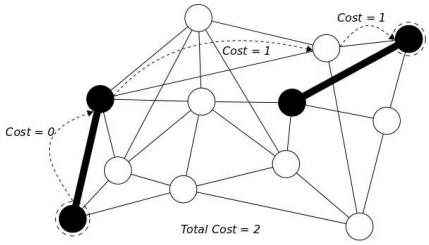


Figure 2: Given a feasible set covering solution, the connectivity evaluation problem *measures* the shortest distance between each selected pair of nodes. The shortest distance between the circled pair of nodes for this solution is 2.

two circled nodes is 2 and the total cost of passing a unit of flow between all selected nodes is 8, since $t > s, \forall (s,t) \in S$.

The mathematical model of the USCCSP is given by,

$$\text{minimise} \quad \sum_{(i,j)\in C} c_{ij}x_{ij} + M \sum_{s\in N} \sum_{t\in T^s} \sum_{(i,j)\in C} z_{ij}^{st}, \tag{1}$$

$$\text{subject to} \quad \sum_{(i,j)\in C} a_{ijq}x_{ij} \geq 1 \quad \forall q \in Q, \tag{2}$$

$$y_{ij}^{st} - x_{ij} \leq z_{ij}^{st} \quad \forall s \in N, \forall t \in T^s, \forall (i,j) \in C, \tag{3}$$

$$\sum_{\substack{i\in N| \\ (i,j)\in C}} y_{ij}^{st} - \sum_{\substack{k\in N| \\ (j,k)\in C}} y_{jk}^{st} = 0 \quad \forall s \in N, \forall t \in T^s, \forall j \in N\backslash\{s,t\}, \tag{4}$$

$$\sum_{\substack{k\in N| \\ (s,k)\in C}} y_{sk}^{st} \geq x_{si} + x_{jt} - 1 \quad \forall s \in N, \forall t \in T^s, \forall (s,i) \in C, \forall (j,t) \in C, \tag{5}$$

$$\sum_{\substack{k\in N| \\ (k,t)\in C}} y_{kt}^{st} \geq x_{si} + x_{jt} - 1 \quad \forall s \in N, \forall t \in T^s, \forall (s,i) \in C, \forall (j,t) \in C, \tag{6}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in C, \tag{7}$$

$$y_{ij}^{st} \in \{0,1\} \quad \forall s \in N, \forall t \in T^s, \forall (i,j) \in C, \tag{8}$$

$$z_{ij}^{st} \in \{0,1\} \quad \forall s \in N, \forall t \in T^s, \forall (i,j) \in C. \tag{9}$$

The solution of the USCCSP minimises the number of separating pairs selected to ensure at least one AA combination is observed in each founder sequence in Q. Furthermore, this problem minimises the number of additional separating pairs that are required for flow to pass between each selected source-sink pair.

The set covering problem of the USCCSP is described by the first term in the objective function (1) and constraints (2). Constraints (2) describe the requirement that each founder sequence $q \in Q$ displays an AA pair contained in at least one selected pair ($x_{ij} = 1$). The connectivity evaluation problem is given by constraints (3)-(6) and the variables y_{ij}^{st} and z_{ij}^{st}. Constraints (3) impose a penalty for the use of each unselected covarying pair (i,j) required for a unit of flow to pass between s and t. The flow balance at each node in the connected path between each (s,t) is given by constraints (4). Constraints (5) and (6) are used to indicate whether a path with nonzero flow must be found between the source-sink pair (s,t), in which case the flow between s and t must be at least one.

4 Solution methods

The USCCSP is a large and complex integer program that is difficult to solve directly using commercial solvers. While each of the individual problems, the set covering and connectivity evaluation problems, can be solved using standard techniques, the integration of the two negatively impacts the problem tractability. Fortunately, the formulation of the USCCSP is particularly suited for the application of Benders' decomposition. This section will describe Benders' decomposition and its application to the USCCSP, including acceleration techniques that are employed.

4.1 Benders' decomposition

The formulation of the USCCSP as the integration of two integer programming problems presents a clear separation of variables permitting the application of Benders' decomposition. Hence, the Benders' master problem (BMP) is described by the set covering problem given by variables x_{ij} and constraints (2). Within the Benders' decomposition framework, a subproblem is formed for each source-sink pair $(s,t) \in S$ describing the connectivity evaluation problem containing the variables y_{ij}^{st} and z_{ij}^{st} and constraints (3)-(6). The result of this decomposition is that the BMP becomes a set covering problem and the subproblems can be solved using a network flow algorithm.

Applying Benders' decomposition to the USCCSP provides a set of separable subproblems that are more easily solved in isolation than as part of the complete problem. The benefit of this separation is seen by fixing the solution values from the BMP, which are then provided as an input to the subproblems. In iteration n, the solution values for the variables in the BMP is given by $\bar{\mathbf{x}}^n = \{\bar{x}_{ij}^n, (i,j) \in C\}$. The solution given by $\bar{\mathbf{x}}^n$ describes a set of separating pairs (i,j) that exhibit at least one feature from each sequence in Q. This set of edges forms a subgraph of the original separating pairs network, which is then passed to the Benders' decomposition subproblem to evaluate the connectivity.

While an individual subproblem is formed for each source-sink pair (s,t), where $s \in N$ and $t \in T^s$, the potentially large number of pairs makes this decomposition computationally impractical. It is deemed more appropriate to formulate one subproblem for each $s \in N$ that is solved to find a connected path from s to every $t \in T^s$. Thus, the primal Benders' subproblem

for source node s and selected pairs $\bar{\mathbf{x}}^n$ (PBSP-s) is given by,

$$\mu_s(\bar{\mathbf{x}}^n) = \text{minimise} \quad \sum_{t \in T^s} \sum_{(i,j) \in C} z_{ij}^{st}, \tag{10}$$

$$\text{subject to} \quad y_{ij}^{st} - z_{ij}^{st} \leq \bar{x}_{ij}^n \quad \forall t \in T^s, \forall (i,j) \in C, \tag{11}$$

$$\sum_{\substack{i \in N| \\ (i,j) \in C}} y_{ij}^{st} - \sum_{\substack{k \in N| \\ (j,k) \in C}} y_{jk}^{st} = 0 \quad \forall t \in T^s, \forall j \in N \backslash \{s,t\}, \tag{12}$$

$$\sum_{\substack{k \in N| \\ (s,k) \in C}} y_{sk}^{st} \geq \bar{x}_{si}^n + \bar{x}_{jt}^n - 1 \quad \forall t \in T^s, \forall (s,i) \in C, \forall (j,t) \in C, \tag{13}$$

$$\sum_{\substack{k \in N| \\ (k,t) \in C}} y_{kt}^{st} \geq \bar{x}_{si}^n + \bar{x}_{jt}^n - 1 \quad \forall t \in T^s, \forall (s,i) \in C, \forall (j,t) \in C, \tag{14}$$

$$y_{ij}^{st} \geq 0, \;\; z_{ij}^{st} \geq 0 \quad \forall t \in T^s, \forall (i,j) \in C. \tag{15}$$

As stated in Section 3, the connectivity evaluation problem determines the fewest number of edges where $x_{ij} = 0$, in a path for a unit of flow to pass between two nodes contained in the selected separating pairs. As such, the connectivity evaluation problem is formulated as a network flow problem, where the source-sink pairs are given by the nodes contained in the set cover. While the PBSP-s does not display the classical form of a network flow problem, it is possible to demonstrate the similarities between these two problems by performing a few simple modifications.

The modifications of the PBSP-s rely on defining a fixed amount of flow through the network and using this to set the cost for passing along each edge. First, the PBSP-s is formulated to identify a path from a single source node $s \in N$ to each of the sink nodes $t \in T^s$. Since the paths for each source-sink pair are independent, the PBSP-s is separable by t and an individual network flow problem can be formulated for each $(s,t), t \in T^s$. Second, Section 3 states that it is only necessary to identify a path between the source-sink pair (s,t) if there exists $i,j \in N, \bar{x}_{si}^n + \bar{x}_{jt}^n - 1 > 0$. This is enforced by the constraints (13) and (14) indicating the amount of flow, $\psi^{st} = \max_{i,j \in N}\{\bar{x}_{si}^n + \bar{x}_{jt}^n - 1\}$, that must pass along the connected path between (s,t). Finally, constraints (11) set the cost of pushing ψ^{st} amount of flow along edge (i,j). Since $\bar{\mathbf{x}}^n$ is set by the solution to the BMP and we aim to achieve the minimum of the total flow, the amount of flow and hence the cost of that flow passing along each edge in the network is fixed.

Using the observations presented above, a classical network flow problem can be formed.

Most importantly, it is possible to eliminate constraints (11) since the cost of pushing ψ^{st} amount of flow along edge (i, j) is fixed by the variables $\bar{\mathbf{x}}$. Using this fixed amount of flow, the variable mapping $z_{ij}^{st} = y_{ij}^{st} \times \max\{\psi^{st} - \bar{x}_{ij}^{n}, 0\}$ can be applied, forming the modified problem. Implementing these modifications results in the formulation of a classical network flow problem that can be efficiently solved using a variety of dedicated solution algorithms. Examples of appropriate network flow algorithms are described in Ahuja *et al.* [1].

4.1.1 Generating Benders' cuts

Solving the PBSP-s using a dedicated network flow algorithm significantly improves the efficiency of the Benders' decomposition solution process. However, employing such an algorithm to solve the PBSP-s provides an optimal primal solution but no dual solution. Further, this optimal primal solution is for the modified problem, which needs to be mapped to the original formulation of the PBSP-s in order to generate cuts. Since Benders' optimality cuts are generated using the optimal dual solutions, further work is necessary to extract this information.

To aid the discussion in this section, the dual variable notation will be provided. First, the dual variables for the connection enforcement constraints (11) are described by $\boldsymbol{\lambda}^s = \{\lambda_{ij}^{st}, \forall t \in T^s, \forall (i,j) \in C\}$. The dual variables $\boldsymbol{\alpha}^s = \{\alpha_j^{st}, \forall t \in T^s, \forall j \in N\}$ are defined for the flow balance constraints (12). Finally, the dual variables for the source and sink node enforcement constraints (13) and (14) are given by $\boldsymbol{\delta}^s = \{\delta_{ij}^{st}, \forall t \in T^s, \forall (s,i) \in C, \forall (j,t) \in C\}$ and $\boldsymbol{\gamma}^s = \{\gamma_{ij}^{st}, \forall t \in T^s, \forall (s,i) \in C, \forall (j,t) \in C\}$ respectively.

Making the assumption that the graph formed by N and C is completely connected guarantees that the PBSP-s is feasible for all solutions $\bar{\mathbf{x}}$ and hence only optimality cuts are generated. An important observation from the covariance and separating pairs networks formed using the HIV *env* sequences is that the resulting graphs are completely connected. This is due to the large number of sequences, the high variability in envelope, and the low cut-off covariance value used. Additionally, the data used for the applications presented in Section 1 also satisfy this assumption.

A Benders' optimality cut describes a feasible region extreme point from the dual of the PBSP-s. Since a dedicated solution algorithm is used to identify the optimal primal solution to the PBSP-s, the optimality cut is constructed by examining the reduced costs of the primal variables. There are four main variables types in the PBSP-s, $y_{ij}^{st}, y_{sk}^{st}, y_{kt}^{st}$ and z_{ij}^{st}, with their

respective reduced cost functions given by,

$$\bar{c}_{ij}^{st} = \alpha_i^{st} - \alpha_j^{st} - \lambda_{ij}^{st} \quad \forall (i,j) \in C, \tag{16}$$

$$\bar{c}_{sk}^{st} = \alpha_s^{st} - \alpha_k^{st} - \sum_{(s,i)\in C} \sum_{(j,t)\in C} \delta_{ij}^{st} - \lambda_{sk}^{st} \quad \forall (s,k) \in C, \tag{17}$$

$$\bar{c}_{kt}^{st} = \alpha_k^{st} - \alpha_t^{st} - \sum_{(s,i)\in C} \sum_{(j,t)\in C} \gamma_{ij}^{st} - \lambda_{kt}^{st} \quad \forall (k,t) \in C, \tag{18}$$

$$\bar{d}_{ij}^{st} = 1 + \lambda_{ij}^{st} \quad \forall (i,j) \in C. \tag{19}$$

The solution to the PBSP-s describes a collection of edges (i,j) that form a connected path p for a flow of ψ_{st} to pass from source node s to sink node t. Hence, the variables y_{ij}^{st}, where $(i,j) \in p$, are basic, implying that their reduced costs are zero. Additionally, the variables z_{ij}^{st} are set to one for each $(i,j) \in p$, if $\bar{x}_{ij} = 0$ and $y_{ij}^{st} = 1$ and zero otherwise, as given by the mapping described previously. Thus, the construction of an optimal dual solution to the PBSP-s is performed using the following algorithm.

The algorithm to construct an optimal dual solution initially sets the values for all dual variables to zero. It is easily verified from the formulation of the PBSP-s dual problem that it is possible to construct feasible solutions where the only non-zero dual variables are related to $(i,j) \in p$. This simplifies the dual construction process, permitting all edges $(i,j) \in C \backslash p$ to be ignored. Since the variables z_{ij}^{st} are basic if $\bar{x}_{ij} = 0$ and $y_{ij}^{st} = 1$, traversing through $(i,j) \in p$, the values of λ_{ij}^{st} are set by the reduced cost function (19). Then considering the source node and the connection $(s,k) \in p$, equation (17) states that it is valid to set $\alpha_k^{st} = -\lambda_{sk}^{st}$. It follows that for every connection $(i,j) \in p, j \neq t$ the related dual variables can be equated using (16), hence $\alpha_j^{st} - \alpha_i^{st} = -\lambda_{ij}^{st}$. Finally, for the sink node and the connection $(k,t) \in p$, equation (18) states that the expression $\sum_{(s,i)\in C} \sum_{(j,t)\in C} \gamma_{ij}^{st} - \alpha_k^{st} = \lambda_{kt}^{st}$ is valid. A solution that satisfies this set of equations is given by setting α_k^{st}, for $k \neq t$, equal to the sum of λ_{ij}^{st} for all connections $(i,j) \in p$ from the source node to k. For $k = t$, $\sum_{(s,k)\in C} \sum_{(l,t)\in C} \gamma_{kl}^{st}$ is set to the sum of λ_{ij}^{st} for all connections $(i,j) \in p$. It is permissible to select any $l' \in N$ such that $\gamma_{kl'}^{st} \in \boldsymbol{\gamma^s}$ is positive, provided that $\bar{x}_{l't} = 1$.

The addition of optimality cuts to the BMP also requires the introduction of the variables φ^s for each subproblem $s \in N$. These variables are bounded from below by the added cuts and provide the current lower bound on the objective value of the PBSP-s. The optimality cuts

added to the BMP are of the form,

$$\varphi^s \geq \sum_{t \in T^s} \sum_{(s,i) \in C} \sum_{(j,t) \in C} \left(\delta_{ij}^{st} + \gamma_{ij}^{st}\right)\left(x_{si} + x_{jt} - 1\right) + \sum_{t \in T^s} \sum_{(i,j) \in C} \lambda_{ij}^{st} x_{ij}. \tag{20}$$

The addition of (20) to the BMP restricts the feasible region and increases the current lower bound of the original problem. Cuts are continually added to the BMP until the gap between the upper and lower bounds, given by the solutions to the BMP and PBSP-s, $\forall s \in S$, respectively, reduces to a desired optimality gap.

4.1.2 Benders' decomposition master problem

The BMP is formulated as a set covering problem with the addition of cuts to express the evaluation of the subgraph connectivity. The set Ω^s is introduced as an index set for the cuts added from the PBSP-s for source node s. Each cut generated from the solution to the PBSP-s is indexed by ω.

The BMP is given by,

$$\text{minimise} \quad \Phi = \sum_{(i,j) \in C} c_{ij} x_{ij} + \sum_{s \in N} \varphi^s, \tag{21}$$

$$\text{subject to} \quad \sum_{(i,j) \in C} a_{ijq} x_{ij} \geq 1 \quad \forall q \in Q, \tag{22}$$

$$\varphi^s \geq \sum_{t \in T^s} \sum_{(s,i) \in C} \sum_{(j,t) \in C} \left(\delta_{ij}^{\omega st} + \gamma_{ij}^{\omega st}\right)\left(x_{si} + x_{jt} - 1\right) + \sum_{t \in T^s} \sum_{(i,j) \in C} \lambda_{ij}^{\omega st} x_{ij} \quad \forall s \in N, \forall \omega \in \Omega^s, \tag{23}$$

$$x_{ij} \in \{0,1\} \ \ \forall (i,j) \in C \quad \varphi^s \in \mathbb{R} \ \ \forall s \in S. \tag{24}$$

The objective value of the BMP in a given iteration provides a lower bound on the optimal solution to the USCCSP. In particular, the values of x_{ij} describe the best selection of separating pairs with the smallest "distance" between each of the identified nodes given the current evaluation information from the PBSP-s, $\forall s \in S$. Since the Benders' decomposition subproblems are always feasible, the solution to the BMP in each iteration is a feasible separating pairs solution. Iteratively solving the BMP and PBSP-s aims to identify a completely connected separating pairs solution.

The variables φ^s provide a lower bound on the objective function value for the PBSP-s and the upper bound is provided by $\mu_s(\bar{\mathbf{x}}^n)$ for a fixed solution to the BMP. Using the upper and

lower bounds, the need for additional optimality cuts from subproblem PBSP-s in the current iteration is identified by $\mu_s(\bar{\mathbf{x}}) > \varphi^s$. The optimal solution to the USCCSP is found when $\mu_s(\bar{\mathbf{x}}) \leq \varphi^s, \forall s \in N$ in a single iteration.

4.2 Acceleration techniques

While Benders' decomposition improves the tractability of large scale optimisation problems, an unfavourable aspect of this approach is slow convergence to the optimal solution. Two common explanations given for the slow convergence is the generation of ineffective cuts from the subproblem and large distances between master problem solutions from consecutive iterations. Approaches employed to address these convergence issues are the generation of Pareto optimal cuts [11], implementing a trust region in the master problem [12] and employing local branching [19]. Each of these approaches has been demonstrated to significantly reduce the solution runtimes by improving the convergence of the Benders' decomposition solution process. To improve the solution runtimes of the USCCSP a trust region method has been employed in the master problem to reduce the number of algorithm iterations. Additionally, a heuristic approach to reduce the size of the separating pairs networks is also presented.

4.2.1 Trust region method

The application of a trust region restricts the feasible region of the master problem to identify solutions close to those found in the previous iteration. Examples of this approach include the addition of a regularisation term in the objective as presented by Ruszczyński [20] and the addition of a set of constraints as demonstrated by Linderoth and Wright [10] and Santoso *et al.* [21]. While both approaches have been demonstrated to improve the convergence of the solution process, the latter maintains the mixed integer programming structure of the master problem and hence it is more appropriate for the USCCSP.

The constraints added to form the trust region restrict the distance between solutions from consecutive iterations. Using the notation presented in Section 4.1, the solution values for the Benders' decomposition master problem in iteration n is given by $\bar{\mathbf{x}}^n$. Given the set of solution values $\bar{\mathbf{x}}^n$ it is possible to define $X^n = \{(i,j)|\bar{x}^n_{ij} = 1\}$ to contain the connections (i,j) related to the basic variables. Thus, the implementation of a trust region for the USCCSP involves the

addition of the following constraint to the BMP,

$$\sum_{(i,j)\notin X^{n-1}} x_{ij} + \sum_{(i,j)\in X^{n-1}} (1 - x_{ij}) \leq \Delta. \tag{25}$$

There are two different implementations of a trust region that can be employed using constraint (25). The first implementation models Δ as a constant and the second as a variable. Modelling Δ as a constant places a hard limit on the number of changes permitted in each iteration. This approach is implemented by Santoso *et al.* [21] where it is stated that by using a fixed value of Δ the convergence of the algorithm is not guaranteed. Hence, it is necessary to increase the value of Δ throughout the solution process such that a non-redundant trust region is maintained. Implementing a trust region by modelling Δ as a variable allows more flexibility between master problem iterations, which is controlled by a large objective coefficient. This alternative approach attempts to avoid the difficulties identified in [21] by penalising the changes in the master problem solution that are made between iterations. However, the convergence of the algorithm is still not guaranteed with this implementation. Additional implementation methods are required to ensure the convergence to the optimal solution. Modelling Δ as a constant is employed in the application of the trust region for the USCCSP, while Δ is modelled as a variable in the extension discussed in Section 5.

The application of a trust region to improve the convergence of the Benders' decomposition solution process has the unfortunate consequence of overestimating the optimal solution. This is the result of constraint (25) restricting the feasible region of the BMP, forcing the solution process to identify a local, but not a global optimum. When implementing the trust region using a constant Δ, this issue is simply addressed by increasing its value when a local optimum is obtained. Resolving the master problem with this increased value of Δ permits the selection of a solution that is "further" away than was previously possible.

An alternative strategy to escape a local optimum is required when modelling Δ as a variable. Specifically, the local optima will provide an upper bound on the global optimal solution, with the objective value given by Φ^{UB}. The following constraint can then be added to the BMP to force subsequent solutions to be at least one better than the best current upper bound (provided that the objective value is guaranteed to be integer),

$$\sum_{(i,j)\in C} c_{ij}x_{ij} + \sum_{s\in N} \varphi^s \leq \Phi^{UB} - 1. \tag{26}$$

If the upper bound solution describes a set of separating pairs forming a completely connected

subgraph, this constraint forces the BMP to identify a connected subgraph containing one less separating pair. Hence, the subsequent iterations of the Benders' decomposition solution process attempt to reduce the number of separating pairs in the optimal subgraph.

4.2.2 Heuristic approach - reducing the number of covarying pairs

The separating pairs network consists of covarying pairs that are exhibited by at least one founder sequence. Since the main objective of the USCCSP is to identify a set of separating pairs that potentially point to important features of the HIV *env* sequences, selecting pairs exhibited by a small number of founder sequences may not be ideal. Specifically, the related mutations described by a separating pair exhibited by one or two sequences may not identify a response due to immune system pressure, but through environmental or random processes. As such, it may be favourable to construct a separating pairs network with at least $k > 1$ sequences exhibiting each separating pair. The described network reduction is employed to solve the extension to the USCCSP presented in Section 5.

5 Problem extension

The applications of the connected subgraph problem introduced in Section 1 involve networks exhibiting a single edge between each pair of nodes. However, in the analysis of gene sequences, it is common for numerous AA combinations to be observed on many separating pairs. Thus, identifying particular AA combinations per pair of positions may present more meaningful results than the positions themselves. An approach to achieve this is presented by Murray *et al.* [15] in the analysis of the hepatitis C virus. Of the two problem formulations presented in [15] the second is solved over a separating pairs network with multiple edges between each pair of positions and at most a single AA combination is chosen per pair in the optimal solution. The construction of the separating pairs network with multiple edges between nodes is also presented in an investigation of HIV sequences by Murray *et al.* [14]. This section discusses an extension to the USCCSP that is solved over a network containing multiple edges for each pair of AA positions.

Examples of the two network construction approaches are presented in Figure 3. In Murray *et al.* [15] and Murray *et al.* [14] at most one edge between two nodes can be selected in the set covering solution. The extension considered in this section also applies this restriction, which

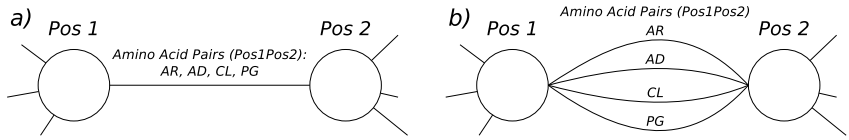


Figure 3: a) a single edge representing all separating covarying amino acid combinations between positions 1 and 2. b) an edge for each separating covarying amino acid combination between positions 1 and 2.

is achieved with the addition of a single set of constraints. However, we must first present the additional notation and variable definitions that are required to describe this extension of the USCCSP, which will be labelled as the USCCSP-AA.

The definitions for the network presented in Section 3 are also employed in the problem formulation given in this section. In addition, for each separating pair $(i, j) \in C$, the set of all observed amino acid combinations (m, n), where AA m is observed at i and n at j, are contained in the set A_{ij}. Extending this definition, the set B_i contains all AA that are observed at position i. For notational convenience, $\hat{C}$ is defined as the set of all separating pairs and related amino acid combinations, given by (i_m, j_n), where $(i, j) \in C$ and $(m, n) \in A_{ij}$.

In the modified network an edge is included for each individual AA combination, hence simply identifying a connected path using these edges may lead to AA mismatches at intermediate nodes. For example, if separating pairs (i, j) and (j, k) are selected with the observed AA combinations (A, R) and (C, L) respectively, the resulting connected subgraph has an AA mismatch at j and will not provide any information about the related mutations between nodes i and k. A more meaningful connected subgraph solution is provided by selecting separating pairs that display the same AA at all common nodes. Using the previous example, a more appropriate subgraph solution would involve the selection of pairs (i, j) and (j, k) with the observed AA combinations (A, R) and (R, C) respectively. This is enforced by the construction of the separating pairs network and the formulation of the connectivity evaluation problem in the USCCSP-AA.

Using the above notation and applying the modified connectivity evaluation, the formulation of the USCCSP-AA is given by,

$$\text{minimise} \quad \sum_{(i_m, j_n) \in \hat{C}} c_{i_m j_n} x_{i_m j_n} + M \sum_{s \in N} \sum_{t \in T^s} \sum_{(i_m, j_n) \in \hat{C}} z^{st}_{i_m j_n}, \tag{27}$$

$$\text{subject to} \quad \sum_{(i_m,j_n)\in\hat{C}} a_{i_m j_n q} x_{i_m j_n} \geq 1 \quad \forall q \in Q, \tag{28}$$

$$\sum_{(m,n)\in A_{ij}} x_{i_m j_n} \leq 1 \quad \forall (i,j) \in C, \tag{29}$$

$$y^{st}_{i_m j_n} - x_{i_m j_n} \leq z^{st}_{i_m j_n} \quad \forall s \in N, \forall t \in T^s, \forall (i_m, j_n) \in \hat{C}, \tag{30}$$

$$\sum_{\substack{i\in N| \\ (i,j)\in C}} \sum_{\substack{l\in B_i| \\ (l,m)\in A_{ij}}} y^{st}_{i_l j_m} - \sum_{\substack{k\in N| \\ (j,k)\in C}} \sum_{\substack{n\in B_k| \\ (m,n)\in A_{jk}}} y^{st}_{j_m k_n} = 0$$

$$\forall s \in N, \forall t \in T^s, \forall j \in N\backslash\{s,t\}, \forall m \in B_j, \tag{31}$$

$$\sum_{\substack{k\in N| \\ (s,k)\in C}} \sum_{(g,h)\in A_{sk}} y^{st}_{s_g k_h} \geq x_{s_m i_n} + x_{j_u t_v} - 1 \quad \forall s \in N, \forall t \in T^s, \forall (s_m, i_n) \in \hat{C}, \forall (j_u, t_v) \in \hat{C}, \tag{32}$$

$$\sum_{\substack{k\in N| \\ (k,t)\in C}} \sum_{(g,h)\in A_{kt}} y^{st}_{k_g t_h} \geq x_{s_m i_n} + x_{j_u t_v} - 1 \quad \forall s \in N, \forall t \in T^s, \forall (s_m, i_n) \in \hat{C}, \forall (j_u, t_v) \in \hat{C}, \tag{33}$$

$$x_{i_m j_n} \in \{0,1\} \quad \forall (i_m, j_n) \in \hat{C}, \tag{34}$$

$$y^{st}_{i_m j_n} \in \{0,1\} \quad \forall s \in N, \forall t \in T^s, \forall (i_m, j_n) \in \hat{C}, \tag{35}$$

$$z^{st}_{i_m j_n} \in \{0,1\} \quad \forall s \in N, \forall t \in T^s, \forall (i_m, j_n) \in \hat{C}. \tag{36}$$

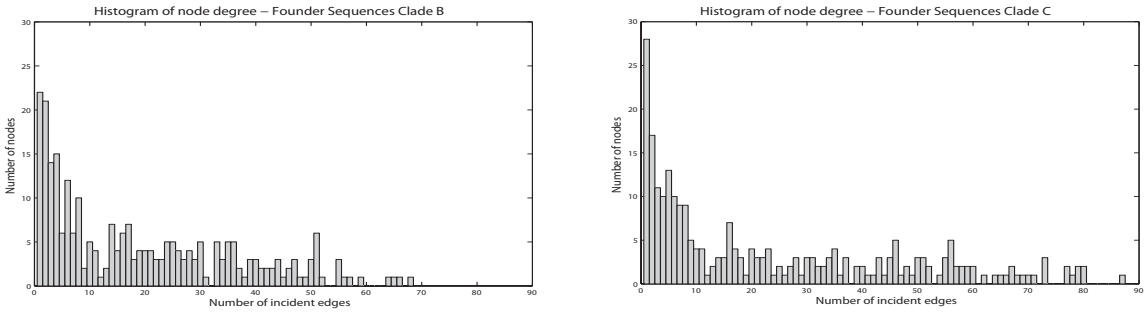
The most important difference between the USCCSP and USCCSP-AA is the addition of constraints (29). This set of constraints is required to ensure that at most one AA combination is selected per pair of positions. The connectivity evaluation constraints (30)-(33) are also modified to properly model the additional edges per pair of positions in the covariance network. In addition, the flow balance constraints (31) are defined for each AA/node combination to ensure the same AA is observed on all edges incoming and outgoing from each node.

The modifications required in response to the alternative network design do not affect the application of Benders' decomposition to the USCCSP-AA. As such, the reformulation of the PBSP-s to a classical network flow problem and the relevant solution algorithms described in Section 4.1.1 can be employed.

6 Computational results

6.1 Solving the USCCSP on the HIV separating pairs networks

The USCCSP is solved to identify a minimum number of covarying pairs forming a completely connected subgraph that separate the founder and chronic HIV *env* sequences. Since each pair of



(a) Clade B. (b) Clade C.

Figure 4: The number of nodes observed to have a given number of edges incident in the separating pairs network.

positions in the network used to solve the USCCSP is connected by a single edge, the resulting connected subgraph provides a broad analysis of the separating pairs. Two different virus subtypes (clades) are investigated in this paper, as such two distinct separating pairs networks are constructed. The USCCSP and USCCSP-AA are solved using Cplex 12.4 interfaced through Matlab 2014a using 12 cores with 15GB RAM.

The data collected for the clade B subtype includes 156 sequences, 78 for both founder and chronic groups. As explained in Section 3, the separating problem of the USCCSP involves selecting covarying pairs that cover the complete set of founder sequences. As such, 78 sequences are contained in the set Q and the resulting separating pairs network consists of 259 nodes and 2495 edges. There are significantly more edges than nodes in the covariance network, resulting in a relatively high degree at each node, which is shown in Figure 4a. Figure 4a demonstrates that a large number of nodes have a small degree, but the majority of nodes have a degree greater than 10. The mean degree for the nodes in the clade B separating pairs network is 19.27 and the median is 15.

Additionally, there are 55 founder and 55 chronic HIV clade C sequences and hence $|Q| = 55$. The clade C separating pairs network consists of 257 nodes and 3021 edges. Figure 4b presents similar node degree results for this separating pairs network, with the same median degree of 15, but a higher mean degree of 23.51.

6.1.1 Clade B separating pairs network - runtime results

The results from solving the BMP without any added cuts is given in the "Separating Pairs Problem" column in Table 1, identifying 6 separating pairs in approximately 0.67 seconds.

The formulation of the USCCSP using the clade B separating pairs network consists of

		USCCSP	
Clade B Separating Pairs Network	Separating Pairs Problem	Standard Benders'	Trust Region
Best Lower Bound	6	8	8
Best Upper Bound	6	-	8
Time to Best Lower Bound (seconds)	0.67	3260.44	4319.73
Time to Best Upper Bound (seconds)	0.67	>36000	50.80

Table 1: The best upper and lower bounds and the time to identify each for the clade B separating pairs network using the standard Benders' decomposition implementation and a trust region method.

167 million variables and 104 million constraints. The vast majority of the variables and constraints are related to the connectivity evaluation. Given the large number of variables and constraints, it is expected that the use of decomposition techniques will greatly improve the problem tractability.

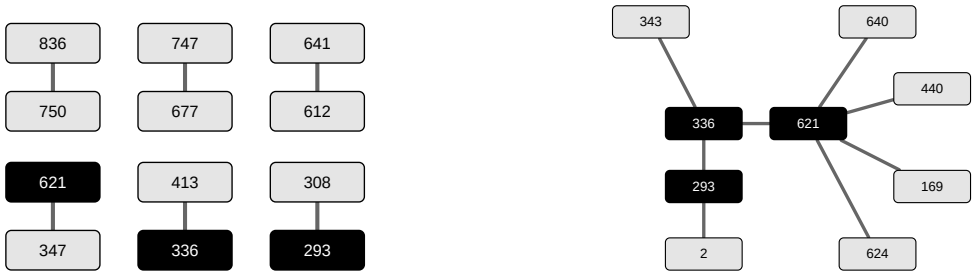
Table 1 presents the runtime results from solving the separating pairs problem and the USCCSP using a standard Benders' decomposition implementation and employing a trust region for clade B. The requirement of identifying a connected subgraph including only the selected separating pairs significantly increases the solution runtime compared to the separating pairs problem of [14]. The standard implementation of Benders' decomposition is unable to solve the USCCSP within 36000 seconds (10 hours). During the solution process, this implementation establishes a lower bound of 8 separating pairs in 3260.44 seconds, but this set of pairs is not completely connected. Comparatively, the trust region method establishes an upper bound of 8 separating pairs forming a connected subgraph in 50.8 seconds. However, it takes a total of 4319.73 seconds to prove optimality. This demonstrates the ability of the trust region approach to identify upper bounds quickly and the difficulty in achieving a good lower bound.

The results presented in Table 1 demonstrate the issues related to the standard implementation of Benders' decomposition. While the standard implementation of Benders' decomposition identifies the optimal lower bound faster than the trust region approach, it fails to find a valid upper bound in the permitted runtime. As demonstrated by Table 1, the trust region approach identifies a good upper bound very quickly, but tuning methods are required to prove optimality.

6.1.2 Clade B separating pairs network - optimal separating connected subgraph

The solution to the USCCSP for the clade B separating pairs network identifies a connected network with 8 separating pairs. By contrast, the solution of the separating pairs problem contains 6 separating pairs. The networks constructed from the sets of separating pairs are presented in Figure 5. This figure demonstrates the lack of connectivity between the pairs after solving the separating pairs problem of Murray *et al.* [15]. By comparison, Figure 5 presents a completely connected subgraph of the separating pairs network. As stated in Section 2, it is likely that few pairs will be common between the solutions to the separating pairs and connected subgraph problems, which is clearly demonstrated in Figure 5. However, there are three nodes common between the two figures (293, 336, 621), highlighted in black. It is interesting to observe that these three nodes form a connected subgraph of the separating pairs solution presented in Figure 5b.

An interesting observation from Figure 5 is the number of nodes selected in the two solutions. While the connected subgraph presents an increase in the number of selected separating pairs, there is a decrease in the number of selected nodes. This is an important result that aids in focusing of the attention to particular features of the *env* sequence.



(a) Separating Pairs. (b) Connected Subgraph.

Figure 5: The separating pairs and connected subgraph solutions for the clade B separating pairs network.

6.1.3 Clade C separating pairs network - runtime results

While fewer clade C sequences were collected to construct the separating pairs network compared to clade B, the number of variables and constraints are greater with 198 million and 126 million respectively. This increase in problem size is the direct result of a larger number of edges in the separating pairs network and suggests that the many positions in the clade C virus subtype experience related mutations.

		USCCSP	
Clade C Separating Pairs Network	Separating Pairs Problem	Standard Benders'	Trust Region
Optimal Number of Pairs	3	4	4
Time to Best Lower Bound (seconds)	~ 0.06	65.77	69.62
Time to Best Upper Bound (seconds)	~ 0.06	116.78	66.64

Table 2: The best upper and lower bounds and the time to identify each for the clade C separating pairs network using the standard Benders' decomposition implementation and a trust region method.

The runtime results for solving the set covering problem and the USCCSP for the clade C separating pairs network are presented in Table 2. While the runtime results presented in Table 2 are much shorter than those presented in Table 1, the comparative results are similar. In particular, there is a significant difference in the runtime to solve the separating pairs problem and the USCCSP. Further, the improvement in the solution runtime from employing the trust region method is also observed. The separating pairs problem, which involves solving the BMP without any added cuts, is solved to optimality in approximately 0.06 seconds and identifies 3 separating pairs. By contrast, the standard implementation of Benders' decomposition to solve the USCCSP finds the optimal separating pairs connected subgraph solution of 4 pairs in 116.78 seconds. This increased runtime is to be expected, since the complexity of the USCCSP is greatly affected by the inclusion of the connectivity constraints.

While the standard implementation of Benders' decomposition solves the USCCSP in very short runtimes, the results for the clade C separating pairs network still demonstrate the strength of the trust region method. By implementing the trust region for this problem, the first upper bound solution of 5 is found after approximately 5.1 seconds and the provably optimal solution of 4 pairs is given in 69.62 seconds. This represents a significant reduction in the solution runtime for the USCCSP. However, the magnitude of this result should not be extrapolated to problems using different networks, since the use of a trust region does not provide a guarantee of improved lower bounds.

6.1.4 Clade C separating pairs network - optimal separating subgraph

The result graphs from solving the separating pairs problem and the USCCSP for clade C are presented in Figure 6. The separating pairs solution consists of three pairs, with two pairs

connected through node 350. By contrast, the connected subgraph require four pairs. Similar to the clade B solution, three nodes are common between the two graphs presented in Figure 6, but there are no common separating pairs. This is not surprising, since only one additional pair is required to form a connected subgraph and that the optimal connected subgraph solution consists of 4 pairs. In a separating pairs network consisting of 3021 edges, it is likely there exists many subgraphs that solve the USCCSP to optimality.

(a) Separating Pairs. (b) Connected Subgraph.

Figure 6: The separating pairs and connected subgraph solutions for the clade C separating pairs network.

6.2 Solving the USCCSP-AA

The extended problem presented in Section 5 involves selecting a set of separating pairs with only a single AA combination per pair. This problem formulation involves a different network construction compared to that used for the USCCSP. In particular, each of the amino acid combinations observed on a pair of positions must be represented by an individual arc. This reconstruction contains the same number of nodes as the original network, but there is a large increase in the number of edges. Specifically, the number of edges in the multiple amino acid (multi-AA) clade B separating pairs network is 12624 and for clade C there are a total of 15663 edges. This is a significant increase in comparison to the original networks containing 2495 and 3021 edges in the clade B and clade C networks respectively. The increase in the number of edges results in a much larger integer program, requiring additional variables and constraints.

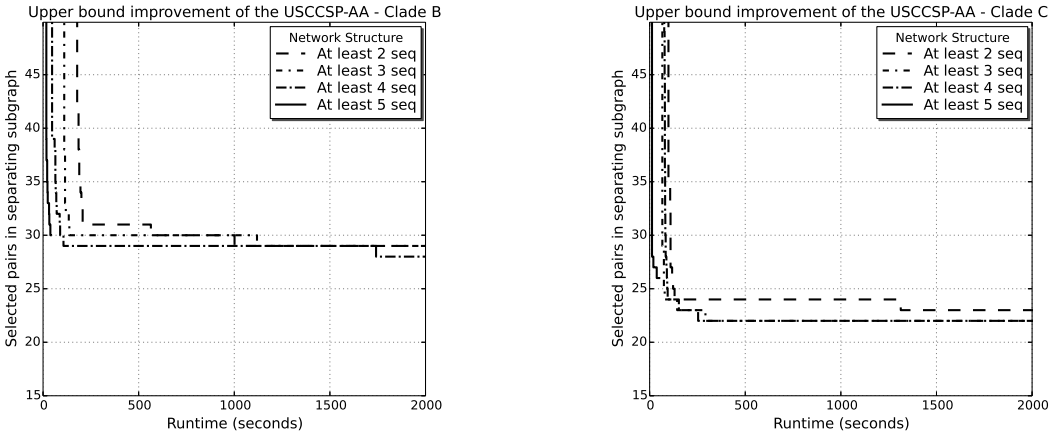
The trust region identifies upper bounds very quickly, but very little improvement in the lower bound is achieved. As such, it is only possible to employ Benders' decomposition as an upper bounding heuristic. However, this heuristic approach still provides meaningful information in the pursuit of identifying a small set of important, connected features that separate founder and chronic sequences.

6.2.1 Improving upper bound solutions with network reductions

Solving the USCCSP-AA using a standard Benders' decomposition implementation is computationally impractical. Given a maximum runtime of 36000 seconds to solve the USCCSP-AA, for the clade B and C separating pairs networks, it is only possible to identify lower bound solutions, with no guarantee of connectivity between the selected pairs. Additionally, from experiments applying a trust region method in isolation, it is observed that while upper bound solutions are identified, large runtimes are required and these solutions may be far from optimal. More specifically, the best found upper bound solutions are 30 and 21 for clade B and C respectively, which are found in 7,684 and 1,353 seconds. The runtime required for the USCCSP-AA is much greater than that presented for the USCCSP in Tables 1 and 2. Hence, the use of additional acceleration techniques to complement the trust region method is beneficial in regards to solution runtimes.

Experiments performed by applying the trust region and network reduction acceleration techniques are presented in Figure 7. In this figure, the labels "At least k seq" indicates that the USCCSP-AA is solved using a network formed by only including the separating pairs with AA combinations that are exhibited by at least k founder sequences.

The presentation of the results in Figure 7 are truncated in the maximum upper bound and solution runtimes. The maximum solution runtime for the USCCSP-AA is set to 36000 seconds. The only experiments with bound improvements after 2000 seconds is when $k = 2$ for both clades. Hence, to improve the presentation, the time axes in Figure 7 are truncated at



(a) Clade B. (b) Clade C.

Figure 7: The improvement in the upper bound for the USCCSP-AA employing a trust region method and network reduction.

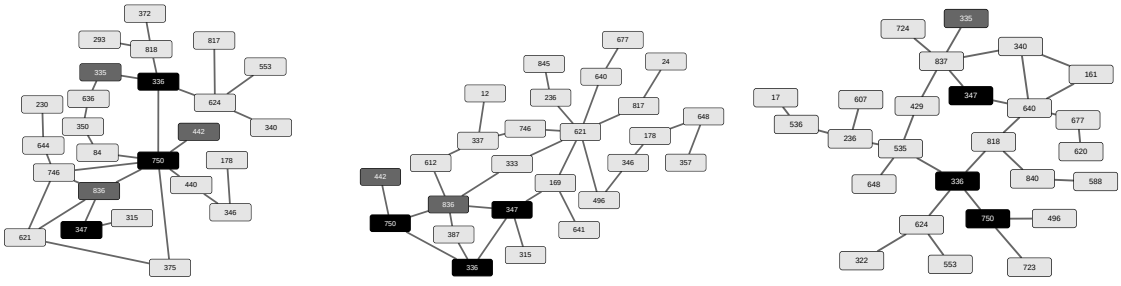
2000 seconds.

The most important result observed in Figure 7 is that the best found upper bound is achieved much faster and in the case of Figure 7a the bound is less than that achieved by solving the complete problem. This clearly demonstrates the benefit from intelligently reducing the number of edges included in the separating pairs network. By comparison, the clade B upper bounds for all network reductions presented in Figure 7a are at most 30. Similarly, the best found upper bound for clade C in Figure 7b is 21, given by the experiment with a minimum of 2 sequences per edge. However, in the case where $k = 2$ for clade C, the runtime required to identify the best found upper bound is greater than for the complete problem. This demonstrates a feature of the trust region approach, where it is difficult for the solution process to escape a local minimum solution.

An interesting observation from Figure 7 is that increasing k results in a faster initial decrease in the upper bound. In particular, each of the experiments identify an upper bound within 3 pairs of their best upper bound in less than 250 seconds. The long tail that follows this initial upper bound improvement further demonstrates the difficulty associated with escaping local optimal solutions when using the trust region method.

6.2.2 Analysis of the separating pairs networks

Figure 7 presents four experiments performed for each clade using different settings for the network reduction. Each of these experiments provide an upper bound solution that are a connected subgraph of separating pairs. Three of the solutions for the experiments presented in Figure 7a are given in Figure 8. The best found upper bound solution is given by networks

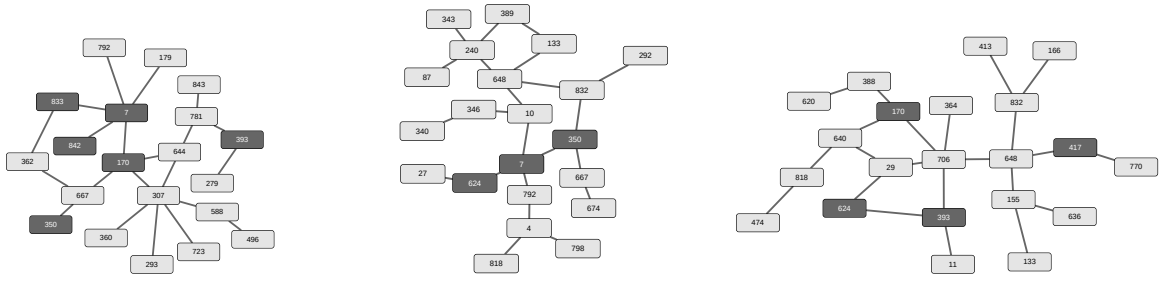


(a) At least 2 sequences. (b) At least 3 sequences. (c) At least 4 sequences.

Figure 8: The best found separating pairs subgraphs for clade B using different network reduction parameters. Black nodes are selected in all of the clade B experiments presented in Figure 7a and similarly the dark grey nodes are selected in all except one experiment.

constructed with a minimum of 2 or 4 sequences exhibited per edge, given in Figures 8a and 8c. It is observed in Figure 8 that the connectivity of the separating pairs subgraph is very high. In these figures there are a number of nodes that have a high degree, many of which are present in the three networks presented. The nodes of particular interest are 336, 347 and 750, which are observed in all upperbounding networks generated for the clade B separating pairs network. In Figure 8 these nodes are highlighted in black and for all networks in Figure 8 the pair (336, 750) is selected as a separating pair. This connectedness suggests that these separating pairs may be of importance to the structure of the HIV *env* sequences, providing a feature for further investigation. Expanding the analysis of the commonly selected nodes, the nodes that appear in at least four of the generated networks are highlighted with dark grey. Two grey highlighted nodes of interest are 442 and 836. In Figures 8a and 8b, these two nodes in conjunction with the black nodes form smaller connected subnetworks. This feature of the separating pairs connected subgraphs potentially identifies other important features of the clade B HIV sequence.

Similar results are observed in the separating pairs subgraph solutions for the clade C separating pairs network, which are presented in Figure 9. The degree of many of the nodes in the subgraphs is high, indicating the importance of particular AA positions. For the clade C separating pairs network, there are no nodes that are selected in all of the experiments presented in Figure 7. This could indicate a much more diverse spread of mutations to the clade C *env* sequence compared to clade B. However, there are 8 nodes, 7, 170, 350, 393, 417, 624, 833 and 842, that are selected in three of the four experiments. In Figure 9 these nodes are highlighted with dark grey, with the majority observed in Figure 9a. While each of the nodes are identified



(a) At least 2 sequences. (b) At least 3 sequences. (c) At least 4 sequences.

Figure 9: The best found separating pairs subgraphs for clade C using different network reduction parameters. The dark grey nodes are selected in all except one of the clade C experiments presented in Figure 7b.

in three of the four experiments, they are not well connected in the resulting subgraphs. This could also be explained by the variability of the clade C HIV sequence. Further investigation of the clade C HIV sequences is necessary to identify a more focused set of AA positions.

7 Conclusions

This paper presents a novel application of operations research to the analysis of HIV *env* sequences. The analysis attempts to construct a connected subgraph of covarying amino acid positions to identify key features related to the transmission of the virus. Two different mathematical models are presented, the second providing a more detailed view of the HIV *env* sequences and related separating pairs networks. A critical feature of this analysis is the connectivity requirement between the selected covarying pairs. This requirement is modelled using a network flow problem, which is at the expense of producing very large problem formulations. The complexity of the resulting problem formulation is addressed by employing Benders' decomposition, along with enhancement techniques, to efficiently solve the USCCSP and reduce runtimes for the USCCSP-AA.

The results demonstrate the general performance of the Benders' decomposition solution approach to solve the USCCSP and USCCSP-AA. The models presented in this paper may be applied to any application requiring the construction of a set covering connected subgraph, in particular those without any specified terminal or root nodes. As such, these results aim to demonstrate the general performance of the developed solution approach.

This paper discusses the implication of identifying connected subgraphs in comparison to the set covering problem of Murray *et al.* [15] and the different results achieved using various enhancement techniques. It is demonstrated that the solution to the USCCSP and USCCSP-AA may identify important features for further research in the operation research and microbiology context.

There are two key areas for further research regarding the work presented in this paper. In the operations research context, identifying other application areas that require the construction of a set covering connected subgraph would aid the further development of the presented approaches. In addition, the results present the use of Benders' decomposition as a heuristic approach to identify good upper bounds to the USCCSP-AA. This presents an area of further research to identify approaches that improve the solution process for the USCCSP-AA, either

through enhancements of the Benders' decomposition approach or the development of alternative techniques. Finally, the separating pairs connected subgraph results attempt to identify key features of the HIV *env* sequence as possible vaccine targets. Further analysis of these connected subgraphs may yield more important features to aid the development of a vaccine preventing the transmission of HIV.

Acknowledgements

We thank Damian Purcell and Talia Mota for collecting and supplying the HIV sequence data.

References

[1] R. Ahuja, T. Magnanti, and J. Orlin. *Network flows: theory, algorithms, and applications*. Prentice Hall, 1993.

[2] E. Álvarez Miranda, I. Ljubić, and P. Mutzel. The rooted maximum node-weight connected subgraph problem. In C. Gomes and M. Sellmann, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 7874 of *Lecture Notes in Computer Science*, pages 300–315. Springer Berlin Heidelberg, 2013.

[3] R. Aurora, M. Donlin, N. Cannon, J. Tavis, et al. Genome-wide hepatitis C virus amino acid covariance networks can predict response to antiviral therapy in humans. *The Journal of clinical investigation*, 119(1):225–236, 2009.

[4] R. Carvajal, M. Constantino, M. Goycoolea, J. P. Vielma, and A. Weintraub. Imposing connectivity constraints in forest planning models. *Operations Research*, 61(4):824–836, 2013.

[5] J. M. Conrad, C. P. Gomes, W.-J. van Hoeve, A. Sabharwal, and J. F. Suter. Connections in networks: hardness of feasibility versus optimality. In P. Hentenryck and L. Wolsey, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 4510 of *Lecture Notes in Computer Science*, pages 16–28. Springer Berlin Heidelberg, 2007.

[6] J. M. Conrad, C. P. Gomes, W.-J. van Hoeve, A. Sabharwal, and J. F. Suter. Wildlife corridors as a connected subgraph problem. *Journal of Environmental Economics and Management*, 63(1):1–18, 2012.

[7] B. Dilkina and C. P. Gomes. Solving connected subgraph problems in wildlife conservation. In A. Lodi, M. Milano, and P. Toth, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 6140 of *Lecture Notes in Computer Science*, pages 102–116. Springer Berlin Heidelberg, 2010.

[8] S. Gnanakaran, T. Bhattacharya, M. Daniels, B. F. Keele, P. T. Hraber, A. S. Lapedes, T. Shen, B. Gaschen, M. Krishnamoorthy, H. Li, J. M. Decker, J. F. Salazar-Gonzalez, S. Wang, C. Jiang, F. Gao, R. Swanstrom, J. A. Anderson, L.-H. Ping, M. S. Cohen, M. Markowitz, P. A. Goepfert, M. S. Saag, J. J. Eron, C. B. Hicks, W. A. Blattner, G. D. Tomaras, M. Asmal, N. L. Letvin, P. B. Gilbert, A. C. DeCamp, C. A. Magaret, W. R. Schief, Y.-E. A. Ban, M. Zhang, K. A. Soderberg, J. G. Sodroski, B. F. Haynes, G. M. Shaw, B. H. Hahn, and B. Korber. Recurrent signature patterns in HIV-1 B clade envelope glycoproteins associated with either early or chronic infections. *PLoS Pathog*, 7(9):e1002209, 2011.

[9] C. P. Gomes, W.-J. van Hoeve, and A. Sabharwal. Connections in networks: a hybrid approach. In L. Perron and M. Trick, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 5015 of *Lecture Notes in Computer Science*, pages 303–307. Springer Berlin Heidelberg, 2008.

[10] J. Linderoth and S. Wright. Decomposition algorithms for stochastic programming on a computational grid. *Computational Optimization and Applications*, 24(2-3):207–250, 2003.

[11] T. Magnanti and R. Wong. Accelerating Benders' decomposition: algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484, 1981.

[12] S. J. Maher, G. Desaulniers, and F. Soumis. Recoverable robust single day aircraft maintenance routing problem. *Computers & Operations Research*, 2014. In press.

[13] T. M. Mota, J. M. Murray, R. J. Center, D. F. J. Purcell, and J. M. McCaw. Application of a case-control study design to investigate genotypic signatures of HIV-1 transmission. *Retrovirology*, 9(1):1–12, 2012.

[14] J. M. Murray, S. J. Maher, T. M. Mota, R. J. Center, and D. F. J. Purcell. Optimal covarying subnetwork differentiating founder and chronic HIV sequences. In preparation.

[15] J. M. Murray, R. Moenne-Loccoz, A. Velay, F. Habersetzer, M. Doffol, J.-P. Gut, I. Fofana, M. B. Zeisel, F. Stoll-Keller, T. F. Baumert, and E. Schvoerer. Genotype 1 hepatitis C virus envelope features that determine antiviral response assessed through optimal covariance networks. *PLoS ONE*, 8(6):e67254, 2013.

[16] H. Önal and R. A. Briers. Designing a conservation reserve network with minimal fragmentation: a linear integer programming approach. *Environmental Modeling & Assessment*, 10(3):193–202, 2005.

[17] H. Önal and R. A. Briers. Optimal selection of a connected reserve network. *Operations Research*, 54(2):379–388, 2006.

[18] H. Önal and Y. Wang. A graph theory approach for designing conservation reserve networks with minimal fragmentation. *Networks*, 51(2):142–152, 2008.

[19] W. Rei, J.-F. Cordeau, M. Gendreau, and P. Soriano. Accelerating Benders' decomposition by local branching. *INFORMS Journal on Computing*, 21(2):333–345, 2009.

[20] A. Ruszczyński. A regularized decomposition method for minimizing a sum of polyhedral functions. *Mathematical Programming*, 35(3):309–333, 1986.

[21] T. Santoso, S. Ahmed, M. Goetschalckx, and A. Shapiro. A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167(1):96–115, 2005.