공학석사학위논문

Flow network model for detection and quantification of gene fusion

유전자 융합 검지, 측량을 위한 플로우 네트워크 모델

2014 년 8 월

서울대학교 대학원

협동과정 계산과학전공

고 세 윤

Flow network model for detection and quantification of gene fusion

유전자 융합 검지, 측량을 위한 플로우 네트워크 모델

지도교수 김 선

이 논문을 공학석사 학위논문으로 제출함

2014 년 6 월

서울대학교 대학원

협동과정 계산과학전공

고 세 윤

고 세 윤의 공학석사 학위논문을 인준함

2014 년 6 월

위 원 장	신 동 우	(인)
부위원장	김 선	(인)
위 원	박 근 수	(인)

Abstract

Flow network model for detection and quantification of gene fusion

Ko Seyoon
Interdisciplinary Program in
Computational Science and Technology
The Graduate School
Seoul National University

Gene fusion is a phenomenon known to have an important role in tumour cells. Tumour heterogeneity is the term that describes how tumour cells have multiple morphologies and phenotypes including gene fusion. As tumour heterogeneity can be explained by using alternative splicing models, one may model fusion gene transcripts in the same way to interpret tumour heterogeneity. However, it is hard for many alternative splicing tools to compute fusion gene models as they have to enumerate paths from the splicing graph. A rigid filter is necessary in this case. For gene fusion problems, the number of exons to model is doubled, making computation much more complex, and filtering can be deemed too heavy. In this thesis, the research was conducted by using a recent alternative splicing tool that directly models a splicing graph and solves the optimization problem over that splicing graph. By doing this, nothing is filtered out before the solving optimization problem. The splicing graph and coverage of

each exon (node) and junction (arc) are computed based on paired-end RNA sequence data. Then the graph is transformed to a canonical convex min-cost flow problem. Then the flow is decomposed into paths which model transcripts after solving a time-consuming optimization problem using a simple heuristic. The results show that this approach in fact works as a sensitive classifier for fusion candidates with only a few paired-end fragments that support the fusion. The method outperformed TopHat and deFuse when applied as a filtering scheme to Chimerascan, whose fusion candidates have the most false positives, in terms of F_3 score, with slight modification.

Keywords: gene fusion, min-cost flow, alternative splicing

Student Number: 2012-20413

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Gene fusion is a biological event in which multiple genes are put together and transcribed jointly after translocations, interstitial deletions, or inversions. Many of the genes involved in gene fusion in cancer cells are oncogenes, and many gene fusions are found in hematological cancers, sarcomas, and prostate cancers[1]. For example, BCR-ABL gene fusion, also known as "Philadelphia chromosome" [2], is known to be associated with chronic myelogeneous leukemia. TMPRSS2-ERG fusion is a common fusion that exists in 40%-70% of human prostate cancers[3]. Also, AML1-ETO fusion[4] is known to be found in many acute myeloid leukemia(AML) cases. Many researchers have been interested in predicting gene fusion events computationally from RNA-seq data.

Alternative splicing is a term that describes mRNA that can code for a number of different proteins through exons skipped, newly included, extended, or shortened, or through the usage of mutually exclusive exons during its processing. Alternative splicing problem is a problem determining the existence and abundance of transcripts from RNA-seq data. It is formulated in Section

2.1 in a generalized form. Within a tumour cell line there are many cells with various morphologies and phenotypes. This phenomenon is called tumour heterogeneity. One needs to consider intra-tumour heterogeneity in order to fully understand genetic and epigenetic traits of cancer[5, 6]. As alternative splicing is one of major factors in tumour progression[7], one may consider tumour heterogeneity of gene fusions in the context of alternative splicing[8]. Thus, the effect of tumour heterogeneity can be assessed by figuring out multiple isoforms of various cells in tumour cell lines.

1.1 Related Works

Many tools that find gene fusions from the RNA-seq data first try to align paired-end reads, realigning initially unaligned reads to find out which genes are involved in gene fusion. TopHat-Fusion[9], deFuse[10], and Chimerascan[11] are three of the most prominent tools in this field. TopHat-Fusion is an extension of alternative splicing detecting tool, TopHat[12]. TopHat tries to map initially unmapped reads as splicing junction reads by splitting the reads. TopHat-Fusion extension then makes use of those reads to nominate fusions. deFuse uses dynamic programming-based approach to split-align the reads. Chimerascan is a tool that finds fusion genes assuming that all gene fusion events happen on exon boundaries. However, these tools concentrate on sequence around the fusion point, not the global fusion transcripts' abundance.

There also have been multiple approaches to estimate the abundance of different isoforms through alternative splicing models. As fusion genes are no exception to alternative splicing, each fusion gene isoform can be quantified by applying alternative splicing quantifiers. Approaches to estimate the abundance of different isoforms through alternative splicing models include: proba-

bilistic models[13], penalized regression[14, 15], and expectation-maximization algorithm[16]. FusionQ[17] is one of the recent tools that quantifies the abundance of full fusion isoforms. In this case, expectation-maximization algorithm[16] is used to solve the problem. In fact, while most of the alternative splicing techniques make use of splicing graphs, which are graphs that show connectivity between exons after splicing, in order to keep the number of variables of $O(n^2)$, these tools have a serious limitation. It is that they have to enumerate each path on the splicing graph before solving optimization problem which involves heavy computation. They have to reduce the number of variables before the optimization step, and during the process many of the isoform candidates are filtered out. For fusion genes, in particular, it is difficult to know what the isoform structure would be, so it is reasonable to use alternative splicing tools that do not filter the transcriptome before solving the optimization problem. A tool named Traph[18] is a tool that solves the problem directly from the splicing graph using a network flow model, transforming the problem to a convex min-cost flow problem. The flow is decomposed into a few paths after optimization, with some heuristics. However, paired-end sequences are not considered in this tool, and nor are transcript directions.

In this work, a framework is constructed to detect and quantify gene fusions and alternative splicings related to specific gene fusion candidates by extending the network flow modelling for paired-end sequence data and gene fusion, in which transcript direction is important. This tool uses full candidates directly built from the "fusion splicing graph" to consider various possible ways of alternative splicing. Its flow network modelling can quantify the abundance of each whole candidate transcriptome in terms of coverage, and it may even consider exon-breaking fusion events. This method also has enhanced sensitivity to fusion candidates with a lower number of fusion-spanning fragments. Further-

more, one can retrieve multiple possible transcriptome candidates per fusion point to consider effect of tumour heterogeneity.

Chapter 2

Methods

2.1 Problem Definition

The goal of the problem is to compute expression level of isoforms in fusion genes, given RNA-seq data, gene locus annotation, and candidate fusion breakpoints along with the number of supporting spanning fragments as input. That is, path and path weights over the "fusion splicing DAG (directed acyclic graph)" are to be constructed. First, one would parse RNA-seq data and gene locus annotation around each fusion breakpoint, and build a fusion splicing DAG $G = (V, E, w_V, w_E)$ with coverage as weight: $w_V(v) = cov(v)$ for $v \in V$, and arc weights $w_E(u, v) = cov(u, v)$ for $(u, v) \in E$. Using this weighted DAG as a major steppingstone, we are to find a tuple of paths minimizing sum of squared error for each exon and junction. This problem is formulated as follows[18].

Problem 1. ***Transcript cover.*** We are given an arc and node-weighted DAG $G = (V, E, w_V, w_E)$, where w_V is a function of node giving weight to nodes,

and w_E is a function of arc giving weight to arcs. The *transcript cover problem* is to find a tuple $\mathcal{P}$ consisting of source-to-sink paths along with path weight, i.e. expression level, $e(P)$ for $P \in \mathcal{P}$ such that

$$\sum_{v \in V} \left(w_V(v) - \sum_{P \in \mathcal{P}: v \in P} e(P) \right)^2 + \sum_{(u,v) \in E} \left(w_E(u,v) - \sum_{p \in \mathcal{P}: (u,v) \in P} e(P) \right)^2 + \sum_{P \in \mathcal{P}} pen(P, e(P))$$

is minimized, where *pen* is the penalty function for each path and its estimated coverage.

The penalty function is used in order to give parsimonious description of RNA-seq experiment result. A widely used penalty term is

$$pen(P, e(P)) = |e(P)|.$$

This means Lasso penalty in the context of regression. This is used in [14] and [15]. For now, as the flow can be decomposed with a few paths after solving the optimization problem, the resulting flow itself does not have to be sparse. Thus, the penalty used here is zero.

Solving Problem 1 over the fusion splicing DAG would give the list of isoforms and coverages. Those isoforms may or may not include the junction representing the fusion breakpoint. If some of the paths include the fusion breakpoint, it may be said that the fusion transcript exists around that candidate breakpoint. If not, the fusion transcript would not exist or is incomplete.

2.2 Proposed Approach

The pipeline proposed consists of three parts: construction of fusion splicing graph weighted by coverage of each exon and junction, solving convex min-cost flow problem, and decomposition. First, "fusion splicing graph" is coustructed, on which each exon is represented by a node, and each junction is represented

by an arc, along with an additional arc representing the candidate fusion point, which is provided as an input. Then, coverage of each exon and junction is computed and is used as weights for each exon and junction. Splice junction mapping result is parsed in this step. This fusion splicing graph construction is novel in that paired-end sequence can be considered, and we can figure out novel junctions in the process. Then, the weighted fusion splicing DAG is produced. A tuple of paths and their weights can be found on this DAG that minimizes sum of squared error for each exon and junction. The definition of this problem is provided in Problem 1. This weighted DAG is transformed into a standard form of convex min-cost problem formulated in Problem 2 and solved with primal network simplex algorithm implemented in [19]. The resulting flow (see Definition 1) is then decomposed into a few paths using greedy heuristics. The whole process is illustrated in Figure 2.1. This procedure will not only consider expression around fusion breakpoint but also expression of the rest of the exons involved in fusion genes, and reconstruct the full structure of fusion transcript as a whole. Furthermore, this method sensitively chooses fusion candidates with the lower number of supporting fragments by correcting coverage of fusion junction with abundance estimation.

2.3 The Weighted Fusion Splicing Graph

2.3.1 Construction of Fusion Splicing Graph

Splicing graph is a graph in which exon-junction relationship of a gene region is represented. Each node denotes an exon and each arc denotes a junction between two exons created by splicing. In general, this graph is a DAG. Each annotated gene region can be modelled by a splicing graph, and if there are multiple isoforms for the model, the DAG is a non-linear chain.

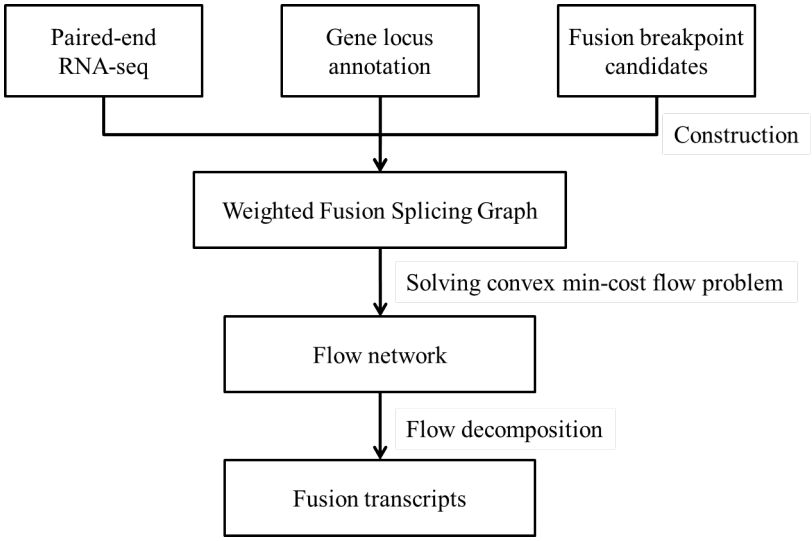

Figure 2.1 The proposed pipeline of fusion transcript detection

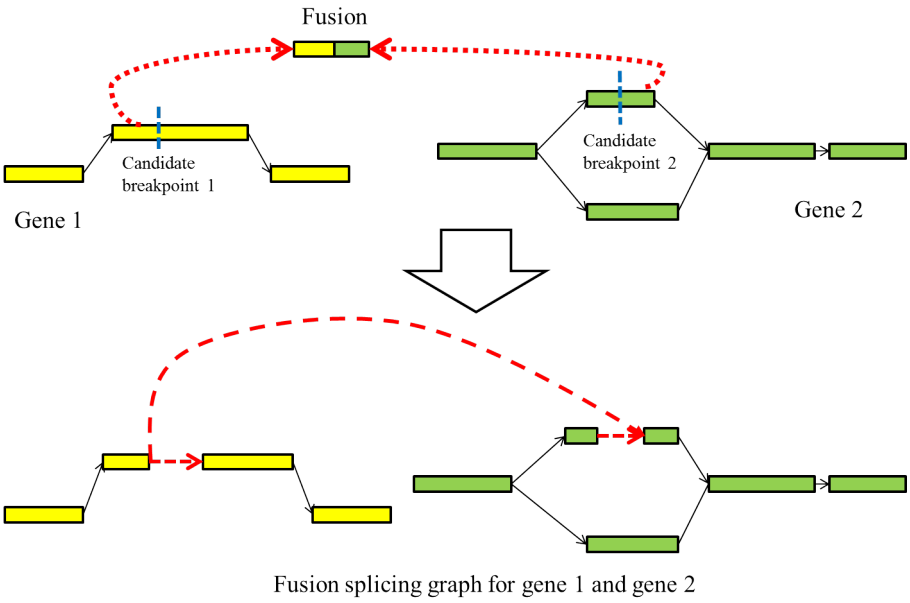

Figure 2.2 An illustration of fusion splicing graph

For the fusion splicing graph, some addition is required. In the simplest case, when the fusion point is exactly at the boundary of exon, an additional arc between two exons is created. However, this is not always the case. When fusion point is on the mid-exon, a node for that exon is broken into two nodes which are connected by a new arc. The two nodes represent the exon regions before and after the breakpoint. This is illustrated in Figure 2.2. Fusion arc from the end of 5'-end to 3'-end is added accordingly. Note that relative transcription direction of each gene region is important, which is not the case when each gene is considered separately. If multiple exons for the region overlap but are not the same, they are broken into non-overlapping nodes as shown in Figure 2.3. Since there is no overlap between the resulting exons, total ordering exists among the nodes.

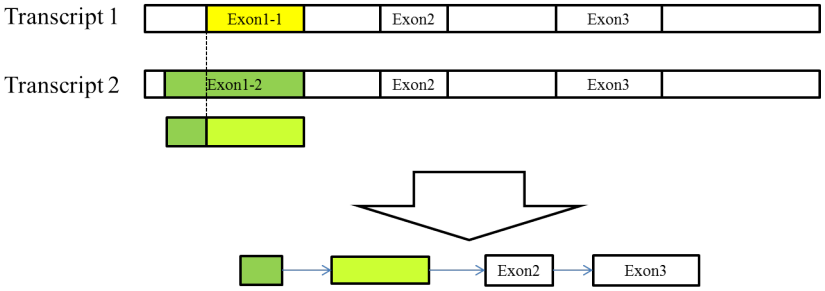

Figure 2.3 Nonoverlapping splicing graph model for overlapping exons

2.3.2 Coverage Computation

Once we construct a fusion splicing graph, the next step is to compute how many times each exon and junction are covered by the RNA-seq data. This process gives weights to nodes and arcs of the fusion splicing DAG. Junction coverage is estimated by the number of junction occurrence in the data. However, since the number of bases each read covers is limited and reads are not uniformly mapped,

exon coverage is computed by the number of total bases covered by the mapping result divided by the length of each "exon" defined in the previous section. De facto standard for sequence mapping with junction splicing is TopHat[12].

In most of gene fusion studies, paired-end sequencing is widely used rather than single-end sequencing due to its possibility that two ends of sequences mapped discordantly help us to find fusion breakpoint. For paired-end sequencing data, estimation of coverage get a little bit complicated. Using Illumina's paired-end sequencing technique, length of paired-end fragment is to be determined empirically, along with standard deviation. In most cases, what is actually sequenced is only a part of the sequence from each end of the fragment. Middle part of the fragment is not sequenced, and the exact length of this part is uncertain. This is illustrated in Figure 2.4. Many models using paired-end sequence model fragment length with normal distribution[15]. In this work, the following scheme is used.

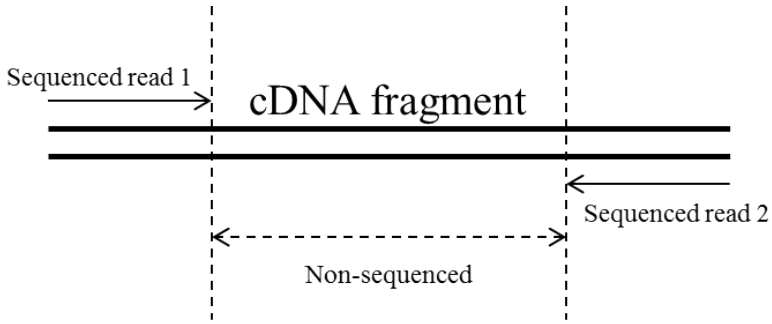

Figure 2.4 Structure of paired-end sequenced cDNA fragment during RNA-seq

First, coverage computation for each exon and junction for the sequenced part is performed. Then, for each mate pair...

- coverage computation for each exon and junction for the sequenced part: directly computed from the mapping result

- coverage computation for non-sequenced part of fragments:
  - If two reads are mapped on the same exon, there is no junction between two ends. The number of bases between two ends mapped is added to exon covering base count. If two reads overlap, this count can be negative (note that sequenced part is already counted).
  - If two reads are mapped on two adjacent nodes, it is clear that there is one junction between two reads. Coverage count for that junction increases by one, and the number of bases between two ends is added to their respective exon covering base count.
  - If two reads are mapped on two distant nodes, one needs to consider all subsets of the set consisting of all nodes between the two mapped nodes. We choose the subset whose sum of length of the members of the set plus non-sequenced region length of two mapped nodes is closest to the mean fragment length minus sequenced length.

The results were combined and normalized by exon length to retrieve the final computed coverage. For fusion junction, coverage is provided externally. If we have non-zero count for a non-annotated pair of nodes from the above procedure, it may be a novel junction or a noise. We decide whether or not to take this as a novel junction based on predetermined threshold value of 2.

2.3.3 Source and Sink Assignment

First, nodes at which an annotated transcript begins were assigned as sources, and nodes at which an annotated transcript ends were assigned as sinks. In addition, if a node has zero in-degree, that node will be also set as sources. Similarly, if a node had zero out-degree, that node was set as sinks.

2.4 Transformation to Min-cost Flow Problem

2.4.1 Definitions of Flow and Convex Min-Cost Flow Problem

Let us discuss the definition of convex min-cost flow problem. Before formulating min-cost flow problem, we present the definition of flow network first. Let us set in-neighbors of $v \in V$ as $I_{G,v} := \{u : (u, v) \in E\}$, and out-neighbors of v as $O_{G,v} := \{u : (v, u) \in E\}$.

Definition 1. ***Flow network and flow.*** A *flow network* is defined by a tuple $N = (G, c, b)$ where $G = (V, E)$ is a directed graph, c assigns a capacity $c_{u,v} \in \mathbb{N}_0$ for each arc $(u, v) \in E$. b defines the supply or demand of each node $v \in V$ depending on the sign. b should meet the **flow conservation** property: $\sum_{v \in V} b_v = 0$. Nodes with $b_v > 0$ are called *sources*, and nodes with $b_v < 0$ are called *sinks*. A *flow* x defined over N is any function of arc $(u, v) \in E$ with $x_{u,v} \in \mathbb{N}_0$ which satisfies the two conditions below:

capacity constraints $0 \leq x_{u,v} \leq c_{u,v} \quad \forall (u, v) \in E$

required flow $\sum_{u \in O_{G,v}} x_{v,u} - \sum_{u \in I_{G,v}} x_{u,v} = b_v \quad \forall v \in V$

Having defined the flow network, the convex min-cost flow problem is defined as follows.

Problem 2. ***Convex min-cost flow problem.*** Given flow network (G, c, b) and a convex cost function $C_{u,v}(x_{u,v}) \geq 0$ for every arc, the *convex min-cost flow problem* is to find a flow x that minimizes $\sum_{(u,v) \in E} C_{u,v}(x_{u,v})$.

The reduction to Problem 2 and decomposing it into a tuple of paths mostly follows the method of Tomescu *et al.*[18], and many definitions and theorems in this thesis are restatements of those stated in Tomescu's paper. It is known

that Problem 2 can be solved in polynomial time using primal network simplex algorithm[20].

2.4.2 Transformation

In Section 2.3, we defined a weighted DAG $G' = (V, E, W_V, W_E)$ of fusion splicing weighted by coverage of each exon and junction. To treat exon coverage and junction coverage equivalently, we need a simple transformation of the graph. For each node $v \in V$, we replace it with two nodes, v_i and v_o, along with a new arc, (v_i, v_o). Also, we set weight to this arc: $w_E(v_i, v_o) = w_V(v)$. In addition, we replace each arc $(u, v) \in E$ with a new arc (u_o, v_i) with $w_E(u_o, v_i) = w_E(u, v)$. Let us call the new network G. For example, the graph on Figure 2.5 (a simplied figure of 2.2 is transformed into Figure 2.6. By this transformation, all weights are now defined on arcs, and node weights were dropped. Now we can easily see that Problem 1 over G' can be reduced to the following problem.

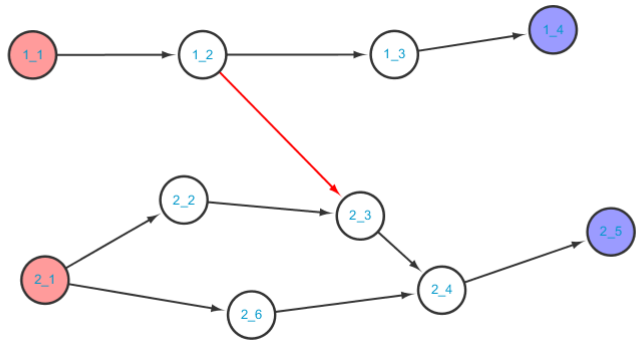

Figure 2.5 Simplified view of Figure 2.2

Problem 3. ***Transcript cover: arc weights only***[18] Given an arc-weighted

DAG $G = (V, E, w_E)$, where w_E is a function of arc giving weight to arcs. The *transcript cover problem for arc weights* is to find a tuple $\mathcal{P}$ consisting of source-to-sink paths along with path weight, $e(P)$ for each $P \in \mathcal{P}$ such that

$$C_{tejc}(\mathcal{P}, e) := \sum_{(u,v)\in E} \left(w_E(u,v) - \sum_{p\in\mathcal{P}:(u,v)\in P} e(P) \right)^2 \tag{2.1}$$

is minimized.

However, it is still impossible to solve the Problem 2 with $N = (G = (V, E), c, b)$ because outward flow from source node $s \in S$ and inward flow to sink note $t \in T$ are not predetermined. Hence, we need to construct an *auxiliary network* in order to reduce the problem to an instance of Problem 2. This auxiliary network $N^* = (G^* = (V^*, E^*), c^*, b^*)$ formulates "offset" of solution from the weight of each arc.

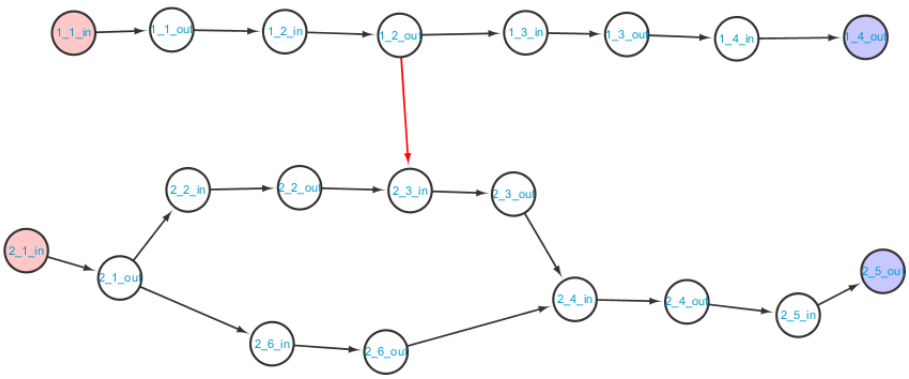

Figure 2.6 Graph of Figure 2.5 transformed

Algorithm 1. Construction of Auxiliary Network

$G^* = (V^*, E^*) \leftarrow G = (V, E)$

$V^* \leftarrow V^* \cup \{s_0, t_0\}$

$b_{s_0} \leftarrow 0; b_{t_0} \leftarrow 0$

for all $s \in S$ **do**

  $E^* \leftarrow E^* \cup \{(s_0, s)\}; c^*_{s_0,s} \leftarrow \infty; C^*_{s_0,s} \leftarrow 0$

end for

for all $t \in T$ **do**

  $E^* \leftarrow E^* \cup \{(t, t_0)\}; c^*_{t,t_0} \leftarrow \infty; C^*_{t,t_0} \leftarrow 0$

end for

$V^* \leftarrow V^* \cup \{s^*, t^*\}; b^*_{s^*} \leftarrow 0; b^*_{t^*} \leftarrow 0$

$E^* \leftarrow E^* \cup \{(t_0, s_0)\}$

for all $(u, v) \in E$ **do** ▷ only for arcs in the original graph

  $c^*_{u,v} \leftarrow \infty; C^*_{u,v}(x) \leftarrow x^2$

  $E^* \leftarrow E^* \cup \{(v, u)\}$ ▷ add reverse arc

  $c^*_{v,u} \leftarrow w_E(u, v); C^*_{v,u}(x) \leftarrow x^2$

end for

for all $v \in V$ **do** ▷ only for nodes from the original graph

  $b^*_v \leftarrow 0$

  if $\sum_{u \in O_{G,v}} w_E(v, u) - \sum_{u \in I_{G,v}} w_E(u, v) > 0$ **then**

    $E^* \leftarrow E^* \cup \{(v, t^*)\}$

    $c^*_{v,t^*} \leftarrow \sum_{u \in O_{G,v}} w_E(v, u) - \sum_{u \in I_{G,v}} w_E(u, v); C^*_{v,t^*} \leftarrow 0$

    $b^*_{t^*} \leftarrow b^*_{t^*} + \sum_{u \in I_{G,v}} w_E(u, v) - \sum_{u \in O_{G,v}} w_E(v, u)$

    ▷ t^* becomes the only sink node of G^*

  else if $\sum_{u \in O_{G,v}} w_E(v, u) - \sum_{u \in I_{G,v}} w_E(u, v) < 0$ **then**

    $E^* \leftarrow E^* \cup \{(s^*, v)\}$

    $c^*_{s^*,v} \leftarrow \sum_{u \in I_{G,v}} w_E(v, u) - \sum u \in O_{G,v} w_E(u, v); C^*_{s^*,v} \leftarrow 0$

    $b^*_{s^*} \leftarrow b^*_{s^*} + \sum_{u \in I_{G,v}} w_E(v, u) - \sum_{u \in O_{G,v}} w_E(u, v)$

    ▷ s^* becomes the only source node of G^*

  end if

end for

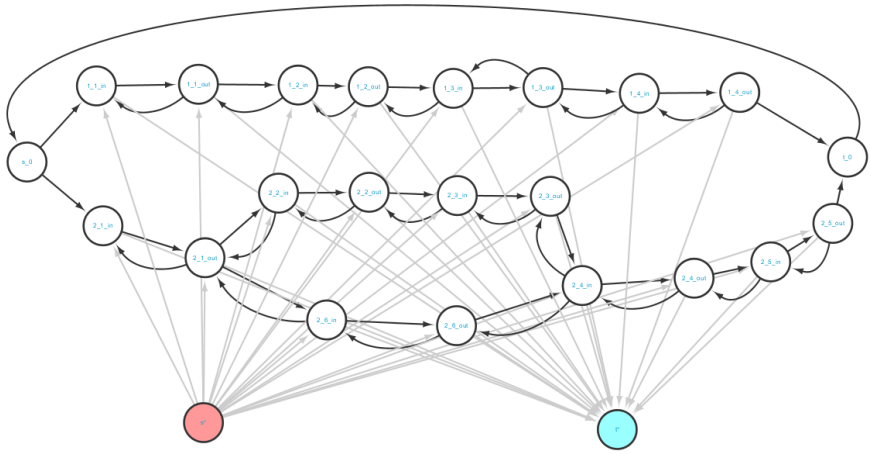

Figure 2.7 Topology of transformed graph for min-cost flow problem. For each vertex, at most one of the grey arcs exist depending on condition

A reverse arc is added for each arc $(u, v) \in E$. A new node s_0 linked to all the original source node $s \in S$ and a node t_0 linked from all the original sink node $t \in T$ are added. Also, a new source node s^* and a new sink node t^* are added. For each node $v \in V$, either arc (s^*, v) or arc (v, t^*) is created, depending on the sign of sum of out-weight minus sum of in-weight of v. The graph on Figure 2.6 is transformed into the topology of 2.7. Grey arcs are included depending on whether total out-weight is bigger than total in-weight or not. The auxiliary flow $x^*_{u,v}$ for $(u, v) \in G$ over G^* means the difference of the resulting "flow" from u to v from the weight of the original network G, and the cost function is simply defined as $C_{u,v}(x) = x^2$. This network specifies exact outward flow from the new source node s^* and inward flow into the new sink node t^*.

Lemma 1. *The auxiliary network $N^* = (G^*, c^*, b^*)$ with cost function C^* built from G in Algorithm 1 is a flow network over which we can solve Problem 2.*

Proof. Since all capacities $c_{u,v}$ were defined as nonnegative value, we only need

to show flow conservation, $\sum_{v \in V^*} b_v = 0$. Furthermore, since s^* is the only source node of G^* and t^* is the only sink node of G^*, we just need to check if $b_{s^*} + b_{t^*} = 0$. Clearly,

$$\begin{aligned} b_{s^*} + b_{t^*} &= \sum_{v \in V} \left(\sum_{u \in I_{G,v}} w_E(u,v) - \sum_{u \in O_{G,v}} (v,u) \right) \\ &= \sum_{(u,v) \in E} w_E(u,v) - \sum_{(v,u) \in E} w_E(v,u) = 0. \end{aligned}$$

□

By Lemma 1, we can now solve Problem 2 over G^*. The following lemma will turn out useful later.

Lemma 2. *At least one of $x^*_{u,v}$ and $x^*_{v,u}$ is zero.*

Proof. If both $x^*_{u,v}$ and $x^*_{v,u}$ were nonzero, a flow y^* with $y^*_{u,v} = x^*_{u,v} - \min(x^*_{u,v}, x^*_{v,u})$, $y^*_{v,u} = x^*_{v,u} - \min x^*_{u,v}, x^*_{v,u}$ and $y^*_{u,v} = x^*_{u,v}$ for all $(u', v') \in E \backslash \{(u,v),(v,u)\}$ would be a flow over N^* with strictly lower cost than x^*. □

After solving the convex min-cost flow problem on N^*, we define a "flow" x over $G = (V, E)$ for each arc $(u, v) \in E$ as follows:

$$x_{u,v} := w_E(u,v) + x^*_{u,v} - x^*_{v,u}. \tag{2.2}$$

If we regard b as a function of x as

$$b_v(x) = \begin{cases} \sum_{u \in O_{G,v}} x_{v,u} \text{ if } v \in S \\ \sum_{u \in I_{G,v}} x_{u,v} \text{ if } v \in T \\ 0 \text{ otherwise} \end{cases}$$

, one may define a flow that meets conditions in Definition 1 over $N = (G, c, b(x))$

with $c = \infty$ and the cost function $C_{u,v}(x) = (x - w_E(u,v))^2$, as long as flow conservation condition is met. Let us see if x is a flow over N.

Theorem 1. *x is a flow over $N = (G, c, b(x))$.*

Proof. We first need to show the flow conservation, and then that x satisfies required flow condition.

Part 1. Flow conservation. It suffices to show that $\sum_{s \in S, v \in O_{G,s}} x_{s,v} = \sum_{t \in T, v \in I_{G,t}} x_{u,t}$. By definition,

$$\sum_{s \in S, v \in O_{G,s}} x_{s,v} = \sum_{s \in S, v \in O_{G,s}} \left(w_E(s,v) + x^*_{s,v} - x^*_{v,s} \right). \tag{2.3}$$

Since $b^*_s = 0$ for $s \in S$ by construction of Algorithm 1 and reqired flow condition on s,

$$x^*_{s,t^*} + \sum_{v \in O_{G,s}} x_{s,v} = \sum_{v \in O_{G,s}} x^*_{v,s} + x^*_{s_0,s}.$$

Thus,

$$\sum_{v \in O_{G,s}} \left(x^*_{s,v} - x^*_{v,s} \right) = x^*_{s_0,s} - x^*_{s,t^*} = x^*_{s_0,s} - b^*_{s,t^*} = x^*_{s_0,s} - \sum_{v \in O_{G,s}} w_E(s,v). \tag{2.4}$$

Plugging (2.4) into (2.3) gives:

$$\sum_{s \in S, v \in O_{G,s}} x_{s,v} = \sum_{s \in S} x^*_{s_0,s} = x^*_{t_0,s_0}. \tag{2.5}$$

Similarly, one may also show that $\sum_{t \in T, v \in I_{G,t}} x_{u,t} = x^*_{t_0,s_0}$.

Part 2. Required flow. Since we have set flow on sources and sinks to meet

required flow condition by construction, it is enough to show that

$$\forall v \in V \backslash (S \cup T), \sum_{u \in I_{G,v}} x_{u,v} = \sum_{u \in O_{G,v}} x_{v,u}. \tag{2.6}$$

By the relationship between x and x^*,

$$\sum_{u \in I_{G,v}} x_{u,v} - \sum_{u \in O_{G,v}} x_{v,u} = \sum_{u \in I_{G,v}} w_E(u,v) - \sum_{u \in O_{G,v}} w_E(v,u) + \sum_{u \in I_{G,v} \cup O_{G,v}} \left(x^*_{u,v} - x^*_{v,u}\right). \tag{2.7}$$

From the construction of Algorithm 1, required outward flows $b^*_{s^*}$ and $b^*_{t^*}$ are defined in a way such that all arcs into t^* and all arcs coming out of s^* require all flow over G^* equal to its capacity. We split this into three cases depending on the sign of $\sum_{u \in I_{G,v} \cup O_{G,v}} \left(x^*_{u,v} - x^*_{v,u}\right)$. First, if $\sum_{u \in I_{G,v} \cup O_{G,v}} \left(x^*_{u,v} - x^*_{v,u}\right) = 0$, it is trivial to see that (2.7) is zero, as $\sum_{u \in I_{G,v}} w_E(u,v) - \sum_{u \in O_{G,v}} w_E(v,u) = 0$. Second, when $\sum_{u \in I_{G,v} \cup O_{G,v}} \left(x^*_{u,v} - x^*_{v,u}\right) > 0$, $\sum_{u \in I_{G,v} \cup O_{G,v}} \left(x^*_{u,v} - x^*_{v,u}\right) = x^*_{v,t^*}$. Because the flow x^* must use full capacity of the arc (v, t^*), $x^*_{v,t^*} = c^*_{v,t^*} = -\left(\sum_{u \in I_{G,v}} w_E(u,v) - \sum_{u \in O_{G,v}} w_E(v,u)\right)$. This proves the claim. One may similarly show that (2.7) holds when $\sum_{u \in I_{G,v} \cup O_{G,v}} \left(x^*_{u,v} - x^*_{v,u}\right) < 0$.

□

2.5 Decomposition of the Flow

It is necessary to decompose the resulting flow into a few paths in order to have a set of transcript models $\mathcal{P}$ paired with their respective coverage $e(P)$ for each $P \in \mathcal{P}$ as the output. It is guaranteed that any decomposition of the solution into a tuple of paths gives the minimum object function for the Problem 1 by the following theorem.

Theorem 2. *Any decomposition of x into a set of tuples of path and its respective expression level results in the same value of objective function for Problem 2. This is also the minimum value of objective function for Problem 3.*

Proof. Any tuple of paths $\mathcal{P}$ from any of $s \in S$ to any of $t \in T$ along with corresponding weights induces a flow over G. This can be done by setting $x_{u,v} = \sum_{P \in \mathcal{P}:(u,v) \in P} e(P)$, and using the similar method for construction of flow network from flow in Section 2.4.2. Because of this, $\min_{\mathcal{P},e} C_{tejc}(\mathcal{P}, e) = \min_x \sum_{(u,v) \in E} C_{u,v}(x_{u,v})$. One may also transform a flow on G^* into a flow on G and vice versa using (2.2). By construction, $\min_x \sum_{(u,v) \in E} C_{u,v}(x_{u,v}) = \min_z \sum_{(u,v) \in E} (z_{u,v} - z_{v,u})^2$. By the fact that $x^* = \operatorname{argmin}_z \sum \left(C^*_{u,v}(z_{u,v}) + C^*_{v,u}(z_{v,u}) \right)$ and Lemma 2, $\min_z \sum_{(u,v) \in E} (z_{u,v} - z_{v,u})^2 = \sum_{(u,v) \in E^*} C^*_{u,v}(x^*_{u,v})$.

□

However, the decomposition of flow into the fewest number of paths is known to be NP-hard in a strong sense[21]. A greedy heuristic is used to decompose the flow into a tuple of paths in reasonable time. The algorithm iteratively removes the heaviest remaining path from source to sink, until there is no such path remaining. This algorithm usually results in only a few paths, helping parsimonious interpretation of gene fusion event from the RNA-seq experiment.

Chapter 3

Results

3.1 Dataset : 40 Reported Gene Fusions

To assess performance of the method as a filter of gene fusion discovery, a public dataset of four breast cancer cell lines is used. These cell lines include 40 known gene fusions reported from [22] and [23]. The data were retrieved from the NCBI Sequence Read Archive under the accession number SRP003186. 21 gene fusions are known in BT-474, 10 in SK-BR-3, 3 in KPL-4 and 6 in MCF-7. The length of each read is 2×50 bp in all cell lines. The fusions are listed in Table 3.1. Empirically estimated mean and standard deviation of fragment length are listed in Table 3.2.

	cell line	5' gene	5' chromosome	3' gene	3' chromosome	source
1	BT-474	ACACA	17	STAC2	17	[22]
2	BT-474	RPS6KB1	17	SNF8	17	[22]
3	BT-474	VAPB	20	IKZF3	17	[22]
4	BT-474	ZMYND8	20	CEP250	20	[22]
5	BT-474	RAB22A	20	MYO9B	19	[22]
6	BT-474	SKA2	17	MYO19	17	[22]
7	BT-474	DIDO1	20	KIAA0406	20	[22]
8	BT-474	STARD3	17	DOK5	20	[22]
9	BT-474	LAMP1	13	MCF2L	13	[22]
10	BT-474	GLB1	3	CMTM7	3	[22]
11	BT-474	CPNE1	20	PI3	20	[22]
12	BT-474	THRA	17	AC090627.1	17	[23]
13	BT-474	TOB1	17	SYNRG	17	[23]
14	BT-474	AHCTF1	1	NAAA	4	[23]
15	BT-474	MED1	17	STXBP4	17	[23]
16	BT-474	MED13	17	BCAS3	17	[23]
17	BT-474	MED1	17	ACSF2	17	[23]
18	BT-474	TRPC4AP	20	MRPL45	17	[23]
19	BT-474	STX16	20	RAE1	20	[23]
20	BT-474	USP32	17	MED1	17	[23]
21	BT-474	PIP4K2B	17	RAD51C	17	[23]
22	SK-BR-3	TATDN1	8	GSDMB	17	[22]
23	SK-BR-3	CSE1L	20	ENSG00000236127	20	[22]
24	SK-BR-3	RARA	17	PKIA	8	[22]
25	SK-BR-3	ANKHD1	5	PCDH1	5	[22]
26	SK-BR-3	CCDC85C	14	SETD3	14	[22]
27	SK-BR-3	SUMF1	3	LRRFIP2	3	[22]
28	SK-BR-3	WDR67	8	ZNF704	8	[22]
29	SK-BR-3	CYTH1	17	EIF3H	8	[22]
30	SK-BR-3	DHX35	20	ITCH	20	[22]
31	SK-BR-3	NFS1	20	PREX1	20	[22]
32	KPL-4	BSG	19	NFIX	19	[22]
33	KPL-4	PPP1R12A	12	SEPT10	2	[22]
34	KPL-4	NOTCH1	9	NUP214	9	[22]
35	MCF-7	BCAS4	20	BCAS3	17	[22]
36	MCF-7	ARFGEF2	20	SULF2	20	[22]
37	MCF-7	RPS6KB1	17	TMEM49	17	[22]
38	MCF-7	GCN1L1	12	MSI1	12	[23]
39	MCF-7	AC099850.1	17	TMEM49	17	[23]
40	MCF-7	SMARCA4	19	CARM1	19	[23]

Table 3.1 List of 40 fusion genes

	mean	standard deviation
BT-474	161	60
SK-BR-3	171	60
KPL-4	150	37
MCF-7	127	44

Table 3.2 Distribution of fragment lengths, including sequenced region

3.2 Fusion Transcript: An Illustration

Let us illustrate an example of fusion transcript evaluation result using RPS6KB1-SNF8 fusion of the cell line BT-474 as an example. RPS6KB1 has 17 nodes (nodes 0-16), and SNF8 has 9 nodes (nodes 17-25). Fusion junction candidate is the arc between node 0 and node 19. Coverage of each exon and junction is illustrated in Figure 3.1. The most-expressed transcript was full transcript of SNF8 with the coverage of 415 (Figure 3.2). Second most expressed transcript was fusion transcript (Figure 3.3). Its coverage was 195. The last isoform was the expression of RPS6KB1 with the coverage of 48 (Figure 3.4).We can carry out relative abundance of fusion transcript compared to original transcript as follows. Of estimated transcripts, exon 0 was covered 243 times, of which fusion junction was covered 195 times following exon 0 (80.2 %). For node 19, it had the coverage of 610, 195 of which followed the fusion junction (32.0 %).

3.3 Assessment of Filtering Performance

First, we compared the performance of three known gene fusion tools, namely, TopHat-Fusion[9], deFuse[10] and Chimerascan[11]. All three tools were initially run with default parameters. TopHat-Fusion and deFuse missed too many reported fusions (15 and 13, respectively), so they were also run with relaxed parameters to increase sensitivity. For TopHat-Fusion, number of minimum fusion reads were decreased to 1 instead of default value of 3. For deFuse, trim length for discordant reads were lowered to 30, rather than default value of 50. The fusion detection results are demonstrated on Table 3.3. Chimerascan was the most sensitive tool even after parameter relaxation of TopHat-Fusion and deFuse.

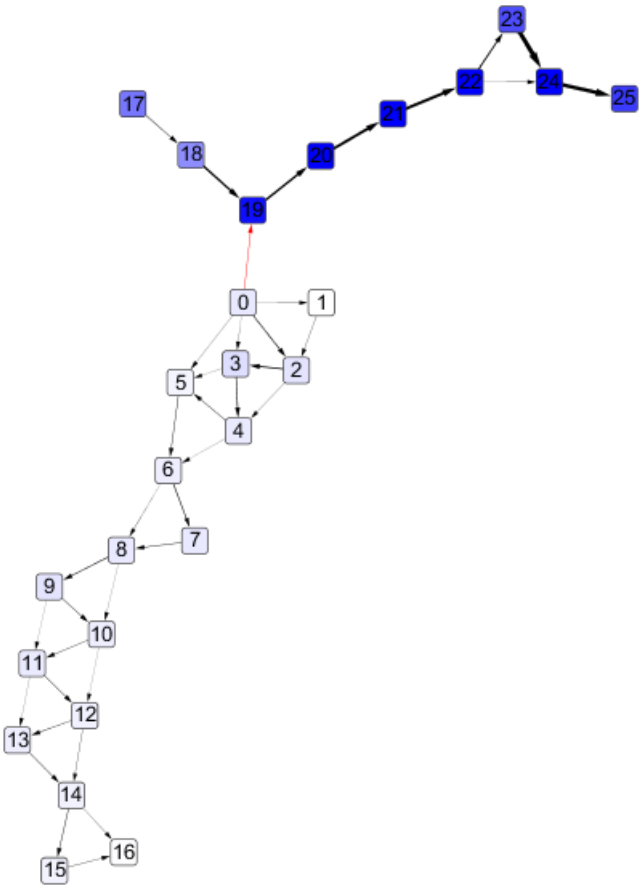

Figure 3.1 Visualization of coverage of nodes and arcs from the fusion RPS6KB1-SNF8 of BT-474. Color of nodes and widths of arc represent coverage.

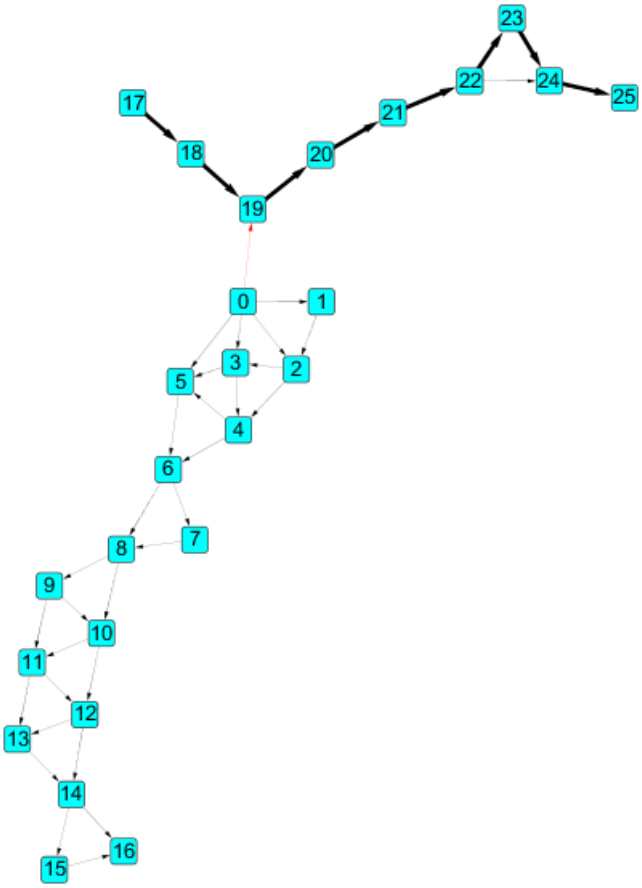

Figure 3.2 1st estimated transcript of RPS6KB1-SNF8, SNF8. coverage=415

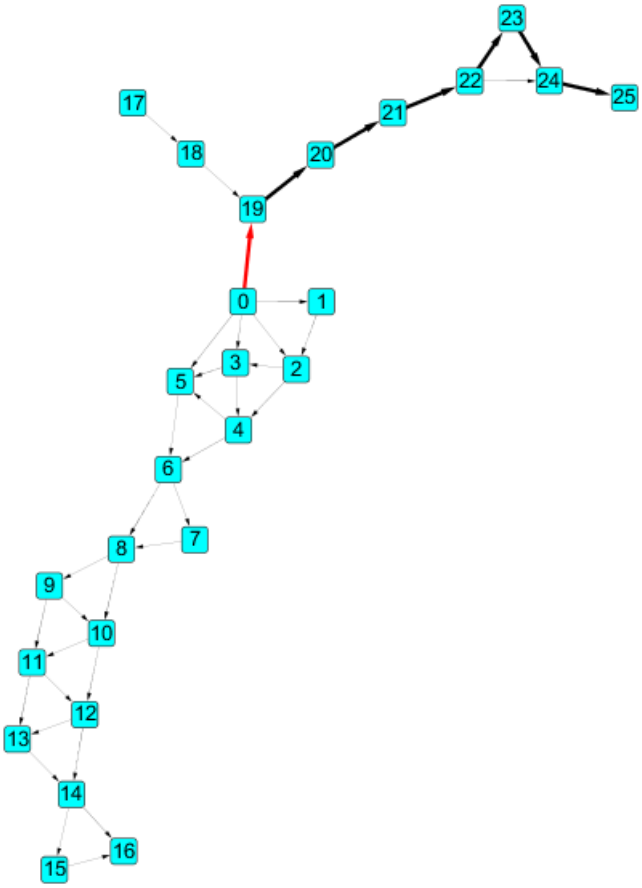

Figure 3.3 2nd estimated transcript of RPS6KB1-SNF8, the fusion transcript, coverage=195

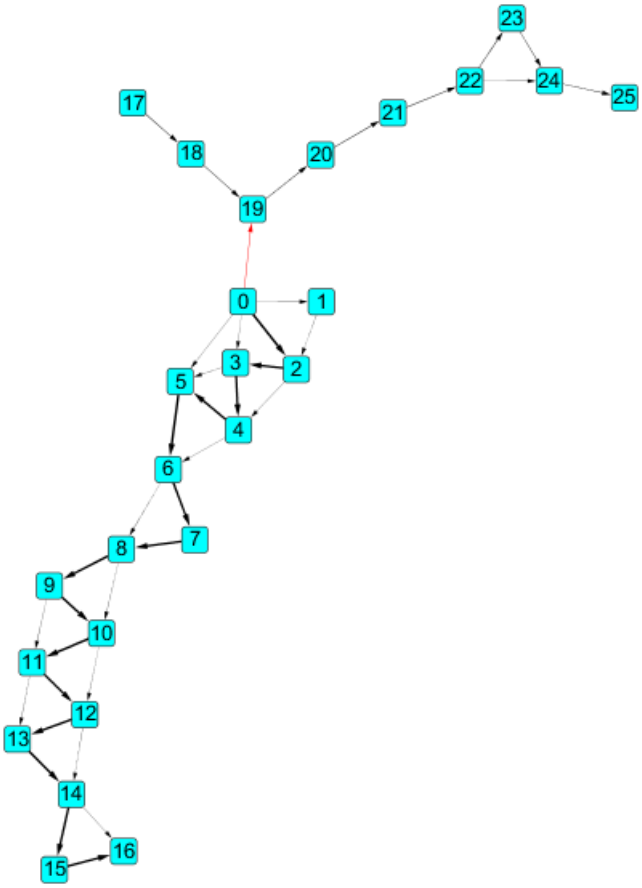

Figure 3.4 3rd estimated transcript of RPS6KB1-SNF8, RPS6KB1. coverage=48

sample	5' gene	3' gene	TopHat Fusion default	TopHat Fusion relaxed	deFuse default	deFuse relaxed	Chimera scan default	Chimera scan top 15%	proposed filtering	proposed saving top 15%
BT-474	ACACA	STAC2	1	1	1	1	1	1	1	1
BT-474	RPS6KB1	SNF8	1	1	1	1	1	1	1	1
BT-474	VAPB	IKZF3	0	1	1	1	1	1	0	1
BT-474	ZMYND8	CEP250	1	1	1	1	1	1	1	1
BT-474	RAB22A	MYO9B	1	1	1	1	1	1	0	1
BT-474	SKA2	MYO19	1	1	1	1	1	1	1	1
BT-474	DIDO1	KIAA0406	0	1	1	1	1	1	1	1
BT-474	STARD3	DOK5	1	1	1	1	1	1	0	1
BT-474	LAMP1	MCF2L	0	1	1	1	1	0	0	0
BT-474	GLB1	CMTM7	0	1	1	1	1	1	1	1
BT-474	CPNE1	PI3	0	1	0	0	1	0	0	0
BT-474	THRA	AC090627.1	0	0	0	1	0	0	0	0
BT-474	TOB1	SYNRG	1	1	1	1	1	1	1	1
BT-474	AHCTF1	NAAA	1	1	1	1	0	0	0	0
BT-474	MED1	STXBP4	1	1	1	1	1	1	1	1
BT-474	MED13	BCAS3	1	1	1	1	1	1	1	1
BT-474	MED1	ACSF2	1	1	1	1	1	1	0	1
BT-474	TRPC4AP	MRPL45	1	1	1	1	0	0	0	0
BT-474	STX16	RAE1	1	1	1	1	1	1	1	1
BT-474	USP32	MED1	0	0	0	0	1	0	1	1
BT-474	PIP4K2B	RAD51C	0	1	1	1	1	0	0	0
BT-474		fusions found	13	19	18	19	18	14	11	15
		reported	46	533	59	114	162	22	57	62
SK-BR-3	TATDN1	GSDMB	1	1	1	1	1	1	1	1
SK-BR-3	CSE1L	ENSG00000236127	0	0	0	0	0	0	0	0
SK-BR-3	RARA	PKIA	1	1	1	1	1	1	0	1
SK-BR-3	ANKHD1	PCDH1	1	1	1	1	1	1	1	1
SK-BR-3	CCDC85C	SETD3	1	1	0	0	1	0	1	1
SK-BR-3	SUMF1	LRRFIP2	1	1	1	1	1	1	1	1
SK-BR-3	WDR67	ZNF704	0	0	0	0	1	0	1	1
SK-BR-3	CYTH1	EIF3H	0	1	1	1	1	1	0	1
SK-BR-3	DHX35	ITCH	0	0	0	1	1	0	1	1
SK-BR-3	NFS1	PREX1	0	0	0	0	0	0	0	0
SK-BR-3		fusions found	5	6	5	6	8	5	6	8
		reported	23	287	38	79	108	15	48	51
KPL-4	BSG	NFIX	1	1	1	1	1	0	1	1
KPL-4	PPP1R12A	SEPT10	1	1	0	1	1	0	1	1
KPL-4	NOTCH1	NUP214	1	1	1	1	1	0	1	1
KPL-4		fusions found	3	3	2	3	3	2	3	3
		reported	7	50	8	16	29	4	7	8
MCF-7	BCAS4	BCAS3	1	1	1	1	1	1	1	1
MCF-7	ARFGEF2	SULF2	1	1	1	1	1	1	1	1
MCF-7	RPS6KB1	TMEM49	1	1	0	1	1	1	1	1
MCF-7	GCN1L1	MSI1	0	0	0	0	1	0	1	1
MCF-7	AC099850.1	TMEM49	1	1	0	1	0	0	0	0
MCF-7	SMARCA4	CARM1	0	1	0	0	1	0	1	1
MCF-7		fusions found	4	5	2	4	5	3	5	5
		reported	12	48	8	28	53	7	23	23
Overall		fusions found	25	33	27	32	34	24	25	31
		reported	88	918	113	237	352	48	135	144

Table 3.3 Fusion detection results of fusion classification methods

Then, the newly-designed method was applied to Chimerascan fusion candidates, as Chimerascan in particular predicts fusion breakpoints on exon boundaries, hence this approach works on it stably. Also, it was the tool with highest number of false positives so that we could demonstrate filtering performance. The newly-developed method filtered out 180 fusion candidates with 8 reported fusion genes in it. Those fusion candidates filtered out sometimes had high number of fragments supporting the fusion, because such candidates usually has high expression of the original gene themselves, and the path through the fusion point is eliminated because of heuristic that takes highest-weighted source-to-sink path, and sometimes one of the two participant gene is never reached by the paths. Thus, top 15% of fusions with highest number of fusion supporting fragments that we could recover gene structure from gene locus annotation were always kept safe from filtering. Then, 13 fusion candidates with 6 reported fusions were saved. This makes the method to have high recall with reasonable precision. To summarize the performance, we used F_β-score as a performance measure. It is weighted harmonic mean of precision= $\frac{TP}{TP+FP}$ and recall= $\frac{TP}{TP+FN}$ defined by:

$$F_\beta = \left(1+\beta^2\right)\frac{\text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}}$$

where β is relative weight of recall compared to precision. In the data, a candidate that is not in the known fusion list may not always mean it is a actual false alarm of gene fusion but may be a previously unreported fusion. In contrast, not reporting a fusion that is in the known fusion list is indeed a mistake made from the tool. Thus, recall is weighted higher in this case, and F_3-score is used as the performance measure. Precisions, recalls, and F_3-scores are all listed on Table 3.4. The modified filtering method scored the highest F_3-score of 0.6151.

This shows sensitive performance of the method on fusion candidates with low number of fusion supporting fragments.

	TopHat Fusion default	TopHat Fusion relaxed	deFuse default	deFuse relaxed	Chimera scan default	Chimera scan top 15%	proposed filtering	proposed saving top 15%
precision	0.2841	0.0359	0.2389	0.1350	0.0966	0.5000	0.1852	0.2153
recall	0.6250	0.8250	0.6750	0.8000	0.8500	0.6000	0.6250	0.7750
F_3-score	0.5580	0.2582	0.5708	0.5360	0.4775	0.5882	0.5051	**0.6151**

Table 3.4 F_3 score of fusion classification methods

Chapter 4

Conculsion

In this work, a framework to detect gene fusions by directly modelling fusion splicing graph using convex min-cost problem formulation is constructed. This tool directly detects fusion transcript based on full candidates that can be built from the splicing DAG. Filtering no fusion candidate from the fusion splicing DAG gives a great deal of flexibility in fusion transcript estimation. Furthermore, it quantifies each transcript in terms of coverage. This can also work as a gene fusion detector/filter. This method can be also used when the fusion point is in mid-exon. One may also see relative abundance of each transcriptome, that is, this may be used to estimate the degree of tumour heterogeneity. This tool is a sensitive classifier for gene fusion event when the number of fragments supporting the fusion is not very high. When applied to Chimerascan results keeping top 15% scorers of it, it had F_3 score of 0.6151, outperforming TopHat, deFuse, or Chimerascan alone. This approach can be extended to search for gene fusions from many cancer samples combinatorially.

Bibliography

[1] F. Mitelman, B. Johansson, and F. Mertens. The impact of translocations and gene fusions on cancer causation. *Nat. Rev. Cancer*, 7(4):233–245, Apr 2007.

[2] A. S. Advani and A. M. Pendergast. Bcr-Abl variants: biological and clinical aspects. *Leuk. Res.*, 26(8):713–720, Aug 2002.

[3] M. A. Rubin, C. A. Maher, and A. M. Chinnaiyan. Common gene rearrangements in prostate cancer. *J. Clin. Oncol.*, 29(27):3659–3668, Sep 2011.

[4] B. Linggi, C. Muller-Tidow, L. van de Locht, M. Hu, J. Nip, H. Serve, W. E. Berdel, B. van der Reijden, D. E. Quelle, J. D. Rowley, J. Cleveland, J. H. Jansen, P. P. Pandolfi, and S. W. Hiebert. The t(8;21) fusion protein, AML1 ETO, specifically represses the transcription of the p14(ARF) tumor suppressor in acute myeloid leukemia. *Nat. Med.*, 8(7):743–750, Jul 2002.

[5] A. Marusyk and K. Polyak. Tumor heterogeneity: causes and consequences. *Biochim. Biophys. Acta*, 1805(1):105–117, Jan 2010.

[6] A. Marusyk, V. Almendro, and K. Polyak. Intra-tumour heterogeneity: a looking glass for cancer? *Nat. Rev. Cancer*, 12(5):323–334, May 2012.

[7] C. Ghigna, C. Valacca, and G. Biamonti. Alternative splicing and tumor progression. *Curr. Genomics*, 9(8):556–570, Dec 2008.

[8] P. Rajan, D. J. Elliott, C. N. Robson, and H. Y. Leung. Alternative splicing and biological heterogeneity in prostate cancer. *Nat Rev Urol*, 6(8):454–460, Aug 2009.

[9] D. Kim and S. L. Salzberg. TopHat-Fusion: an algorithm for discovery of novel fusion transcripts. *Genome Biol.*, 12(8):R72, 2011.

[10] A. McPherson, F. Hormozdiari, A. Zayed, R. Giuliany, G. Ha, M. G. Sun, M. Griffith, A. Heravi Moussavi, J. Senz, N. Melnyk, M. Pacheco, M. A. Marra, M. Hirst, T. O. Nielsen, S. C. Sahinalp, D. Huntsman, and S. P. Shah. deFuse: an algorithm for gene fusion discovery in tumor RNA-Seq data. *PLoS Comput. Biol.*, 7(5):e1001138, May 2011.

[11] M. K. Iyer, A. M. Chinnaiyan, and C. A. Maher. ChimeraScan: a tool for identifying chimeric transcription in sequencing data. *Bioinformatics*, 27(20):2903–2904, Oct 2011.

[12] Cole Trapnell, Lior Pachter, and Steven L. Salzberg. Tophat: discovering splice junctions with rna-seq. *Bioinformatics*, 25(9):1105–1111, 2009.

[13] C. Trapnell, B. A. Williams, G. Pertea, A. Mortazavi, G. Kwan, M. J. van Baren, S. L. Salzberg, B. J. Wold, and L. Pachter. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat. Biotechnol.*, 28(5):511–515, May 2010.

[14] W. Li, J. Feng, and T. Jiang. IsoLasso: a LASSO regression approach to RNA-Seq based transcriptome assembly. *J. Comput. Biol.*, 18(11):1693–1707, Nov 2011.

[15] J. J. Li, C. R. Jiang, J. B. Brown, H. Huang, and P. J. Bickel. Sparse linear modeling of next-generation mRNA sequencing (RNA-Seq) data for isoform discovery and abundance estimation. *Proc. Natl. Acad. Sci. U.S.A.*, 108(50):19867–19872, Dec 2011.

[16] B. Li and C. N. Dewey. RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. *BMC Bioinformatics*, 12:323, 2011.

[17] C. Liu, J. Ma, C. J. Chang, and X. Zhou. FusionQ: a novel approach for gene fusion detection and quantification from paired-end RNA-Seq. *BMC Bioinformatics*, 14:193, 2013.

[18] A. I. Tomescu, A. Kuosmanen, R. Rizzi, and V. Makinen. A novel min-cost flow method for estimating transcript expression with RNA-Seq. *BMC Bioinformatics*, 14 Suppl 5:S15, 2013.

[19] Balázs Dezső, Alpár Jüttner, and Péter Kovács. LEMON – an open source c++ graph template library. *Electronic Notes in Theoretical Computer Science*, 264(5):23 – 45, 2011. Proceedings of the Second Workshop on Generative Technologies (WGT) 2010.

[20] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, Englewood Cliffs, NJ, 1993.

[21] B. Vatinlen, F. Chauvet, P. Chrétienne, and P. Mahey. Simple bounds and greedy algorithms for decomposing a flow into a minimal set of paths. *European Journal of Operational Research*, 185(3):1390 – 1401, 2008.

[22] H. Edgren, A. Murumagi, S. Kangaspeska, D. Nicorici, V. Hongisto, K. Kleivi, I. H. Rye, S. Nyberg, M. Wolf, A. L. Borresen-Dale, and O. Kallioniemi. Identification of fusion genes in breast cancer by paired-end RNA-sequencing. *Genome Biol.*, 12(1):R6, 2011.

[23] S. Kangaspeska, S. Hultsch, H. Edgren, D. Nicorici, A. Murumagi, and O. Kallioniemi. Reanalysis of RNA-sequencing data reveals several additional fusion genes with multiple isoforms. *PLoS ONE*, 7(10):e48745, 2012.

요약

유전자 융합은 암 종양의 발달의 중요한 요인 중 하나이다. 종양 이질성은 종양 세포들 안에 다양한 표현형과 변이가 존재함을 말하는데, 대체 스플라이싱은 이 종양 이질성의 대표적인 예이다. 따라서 유전자 융합의 종양 이질성 연구를 위해 융합 유전자의 대체 스플라이싱을 모델링하여 다양한 융합 유전자의 발현을 확인할 수 있다. 많은 대체 스플라이싱 모델을 활용한 도구들은 스플라이싱 그래프 상의 경로를 일일이 열거해 가며 모델링하기 때문에 변수의 개수가 많아져 많은 수의 후보를 필터링해야 한다. 유전자 두 개의 융합을 모델링하는 문제에서는 단순 대체 스플라이싱 모델에 비해 모델링해야 되는 엑손의 수가 늘어나게 되기 때문에 이러한 방식으로 문제를 해결하기에는 너무나 많은 필터링을 필요로 하게 된다. 본 연구에서는 네트워크 플로우 모델을 이용한 모델링을 활용하여 전사 모델을 필터링하기 전에 최적화 문제를 푼 뒤 휴리스틱을 이용하여 그래프 상의 경로를 찾아 전사 모델을 찾는다. 우선 쌍끝 RNA-서열 데이터를 이용하여 융합 스플라이싱 그래프와 각 엑손(꼭지점), 결합점(변)의 커버리지를 계산한 뒤 이 가중 그래프를 표준적인 볼록 최소-가격 플로우 문제로 변환하여 문제를 풀고, 그 결과 나온 플로우를 휴리스틱을 이용하여 여러 개의 경로로 분해하였다. 각각의 경로는 각각의 전사 모델을 나타내게 된다. 이 방법을 이용하여 융합 유전자 후보군에서 실제 융합 전사체의 존재를 확인할 수 있었다. 또한, 이 방법을 다른 유전자 융합 검지 도구에 비해 가장 많은 위양성 결과를 보여준 Chimerascan에 적용하여 약간의 수정을 거쳐 필터링하는 방법으로 TopHat-Fusion과 deFuse보다 좋은 F_3 점수를 갖는 분류 결과를 얻을 수 있었다.

주요어: 유전자 융합, 플로우 네트워크, 대체 스플라이싱

학번: 2012-20413