BBQ in Tanimoto scores

Peter Menzel

Bled, 2006

Regulation of transcription

- transcription of DNA into RNA is done by RNA polymerases I, II, III
- each polymerase requires *transcription binding factors* which bind to short specific sequences located near the transcription start site, *transcription factor binding sites, TFBS*
- we assume that these TFBS occur clustered, i.e. they form a (upstream) regulatory module, and
- these modules can be found in similar regulated genes
- additionally the modules must not share precisely the same set of TFBS, but share a significant number of common sites

Problem Overview

- given K (upstream) sequences
- m candidate binding sites
- module length L
- we want to find the largest subset of the m binding sites which occur clustered within an interval of length L in each of the K sequences
- we call this largest subset a **best bbq**
- Problem is NP-complete

Setting

- 1st step: identify clusters of length L in each sequence $1..K$
- we obtain for each sequence i a cell set $C_i = \{B_{i,1}, ..., B_{i,\lambda_i}\}$, with $B_{i,j} \subseteq [1 : m]$
- **Instance:** Given $m, K, C_1, ..., C_K$ with $\lambda_i := |C_i|$, maximize

$$\left| \bigcap_{i \in [1:K]} B_{i,\nu_i} \right| \text{ with } \nu_i \in [1 : \lambda_i]$$

Algorithm A1

- for each $(\nu_1, ..., \nu_K) \in [1 : \lambda_1] \times ... \times [1 : \lambda_K]$ compute $|\bigcap_{i \in [1:K]} B_{i,\nu_i}|$
- keep track of the largest cardinality intersection

Algorithm A2

- enumerate all subsets of $[1 : m]$
- for each $A \subseteq [1 : m]$ check whether there are suitable indizes $\nu_1, ..., \nu_K$ such that

$$A \subseteq \bigcap_{i \in [1:K]} B_{i,\nu_i}$$

- keep track of the largest cardinality subset, for which suitable indizes were found.

Complexity

- time complexity of **A1** is $\mathcal{O}(Km\lambda^K)$, with $\lambda = \max_i \lambda_i$
- **A2** is in $\mathcal{O}(2^m \Lambda m)$ with $\Lambda := |C_1| + ... + |C_K|$
- branch-and-bound modifications for both **A1** and **A2** are applicable

- time complexity of **A1** is $\mathcal{O}(Km\lambda^K)$, with $\lambda = \max_i \lambda_i$
- **A2** is in $\mathcal{O}(2^m \Lambda m)$ with $\Lambda := |C_1| + ... + |C_K|$
- branch-and-bound modifications for both **A1** and **A2** are applicable

- complexity of **A2**
  - one set containment test costs $\mathcal{O}(m)$
  - every test of candidate A costs $\mathcal{O}(\Lambda)$
  - we have 2^m many candidates A to test
  - resulting in $\mathcal{O}(2^m \Lambda m)$
- complexity of **A1**
  - one set containment test costs $\mathcal{O}(m)$
  - testing one vector $(\nu_1, ..., \nu_K)$ costs $\mathcal{O}(Km)$
  - we have λ^K many vectors to test
  - resulting in $\mathcal{O}(\lambda^K Km)$

Improving performance

- improvements for **A2**:
  1. if $A \subseteq [1:m]$ is no bbq then all A' with $A \subseteq A'$ are no bbq either
  2. consider only $A \subseteq [1:m]$ such that some superset of A is contained in at least one C_i

Limitations

- original bbq algorithm provides always the best **accurate** results
- the best bbq is always contained in **all** given sequences.
- while it is nice to obtain always correct results, this approach has several drawbacks

Limitations

- success of the original algorithm strongly depends on a careful selection of the input sequences
- if we have only one *bad* sequence, which contains only a few binding sites, the overall result is restricted to the sites contained in this sequence.
- if this particular bad sequence contains no sites at all, the search returns no best bbq, although all other sequences might share a good set of binding sites

Limitations

- let $B_1 = \{1, 2\}, B_2 = \{1, 3\}$ and $B_3 = \{2, 3\}$ be the binding sites contained in sequences 1 to 3.
- the bbq algorithm has no chance to find a best bbq A with $A \subseteq B_1 \wedge A \subseteq B_2 \wedge A \subseteq B_3$, simply because it does not exist.
- using a score-based approach may yield a better result: $A = \{1, 2, 3\}$.
  - although A is not a subset of any B_i, it is a good representation of the sites found in the three sequences.

Scoring function

- we want a scoring function f, which applied to a candidate set A, gives a similarity score *how good* it matches the given arrangement
- this will lead to non-accurate results

Tanimoto scores

- similarity score between two sets X and Y

$$\tan(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

- applied to best bbq problem:
- extend algorithm **A2**, which enumerates candidate sets
- for each $A \subseteq [1 : m]$ calculate

$$\mathrm{T}(A) = \sum_{i=1}^{K} \max_{j=1}^{\lambda_i} \tan(A, B_{i,j})$$

- candidate set with highest $\mathrm{T}(A)$ matches best the given clusters

Tanimoto variants

1. rate of same elements and different elements

$$\tan(X, Y) = \frac{|X \cap Y|}{|X \setminus Y \cup Y \setminus X|} = \frac{|X \cap Y|}{|X \triangle Y|}$$

2. how many of all elements occur in both sets

$$\tan(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

3. increase the weight of elements which occur in both sets

$$tan(X, Y) = \left(\frac{(|X \cap Y|)^2}{|X \cup Y|} \right)^2$$

Limitations

- non-accurate results
- major disadvantage ist the runtime complexity, which depends on number of binding sites tested
- branch-and-bound algorithms are not applicable
- thus, we implemented a so called *bounded differences* version of the algorithm

Bounded differences

- reduce runtime complexity by reducing number of candidates tested
- only consider candidate sets which are similar to the cells $B_{i,j}$
- candidates may only deviate by δ elements from the original cells
- constructing a set of candidates while computing the delta-bounded differences
- time complexity decreased radically, memory consumption increases.
- the cardinality of this set of candidates is the major factor of the memory complexity

Weighted versions

- TFBS are defined by a *position weight matrix, PWM*
- each occurence of a TFBS gets a weight *how strong* the match is
- elements belong to the sets $B_{i,j}$ to *certain degree*: $B = \{0.8/3, 0.7/4, 1/5, 0.4/6\}$
- calculating $\tan(A, B) = \frac{|A \cap B|}{|A \cup B|}$ requires non-standard (fuzzy) set operations

Overlap-free BBQ

- TFBS occuring in one cluster may not overlap
  1. for each cell $B_{i,j}$, calculate overlap graph of its contained sites
  2. construct complement graph
  3. find the maximum clique X
  4. calculate $\tan(A, X)$
- **Problem:** Maximum clique problem is *NP-complete*, resulting in additional runtime complexity