Common Substrings in Random Strings

Eric Blais, Mathieu Blanchette

McGill Centre for Bioinformatics

[eblais,blanchem]@mcb.mcgill.ca

Common *k*-Substring Problem

- **Given**: A set of strings S_1, S_2, …, S_r and a length k,
- **Determine**: If there is a string T of length k that is a substring of each of S_1, S_2, …, S_r

Common *k*-Substring Problem

```
CTTGCTTAGTCTTTTGCTGTGTGCTAATGTTTCGAAATGT
CTCAGCTAATCTGCGAGAGCTTCAGGGGCGACAATCTACG
AGTGCATACAGACTCTAATACCAAATTGTGGAACACCATC
TGGATAGACTATTCCCCGTTATTACCGCTAATGGAGTCGT
CAGGAATATGTATCTCATACCCCGGCTAATAAGGGTTAGA
ACAAAAGGAAGCGGCGGCTTGACGTGTACCTAATGATCGT
TCTAATGGAACGGGGCACTTCGGCTAAATACAGGGAGAGT
TGTCACATCTAATTTCTACCTTACGACCGTCTCGAGTAGC
TAAACTTGCACTAACCTAATTGTCTGGGTTTGGAGAGCG
GGCATGCCCCGATGCTGTGTAGCCCTAATAGACCGGATAA
```

Common *k*-Substring Problem

CTTGCTTAGTCTTTTGCTGTGTGCTAATGTTTCGAAATGT
CTCAGCTAATCTGCGAGAGCTTCAGGGGCGACAATCTACG
AGTGCATACAGACTCTAATACCAAATTGTGGAACACCATC
TGGATAGACTATTCCCCGTTATTACCGCTAATGGAGTCGT
CAGGAATATGTATCTCATACCCCGGCTAATAAGGGTTAGA
ACAAAAGGAAGCGGCGGCTTGACGTGTACCTAATGATCGT
TCTAATGGAACGGGGCACTTCGGCTAAATACAGGGAGAGT
TGTCACATCTAATTTCTACCTTACGACCGTCTCGAGTAGC
TAACTTGCACTAACCTAATTGTCTGGGTTTGGGAGAGCG
GGCATGCCCCGATGCTGTGTAGCCCTAATAGACCGGATAA

Common k-Substring in Random Strings (CSRS) Problem

- **Given**: A random process P that generates set of strings S_1, S_2, …, S_r and a length k,
- **Find**: The probability that there is a string T of length k that is a substring of each of S_1, S_2, …, S_r

History of the CSRS Problem

- Study began 20+ years ago, when Arratia & Waterman examined the asymptotic behavior of the length of the longest perfect alignment between two random strings.
- Results to date offer good approximation when the number of random strings is **low**, but poor approximations when there are many random strings in the problem instance.

New Approximations to the CSRS Problem

- We present 2 new approximations for the CSRS problem, aimed specifically at being accurate when there are many random strings:
  1. Independent Words Model
  2. Double Independence Model

Independent Words Model

- **Independent Words Assumption**: Different k-substrings occur independently of each other in a random string

Independent Words Model

- Resulting approximation:

$$P(\zeta) = 1 - \prod_{w \in \Sigma^k} \left(1 - P(\xi_w)^r\right)$$

- Notation:

$P(\xi_w)$ – Probability of the string *w* occurring in a random string

$P(\zeta)$ – Probability that at least one string of length *k* occurs as a common substring to all the random strings

Independent Words Model

Table 1: Approximations to CSRS for $k = 6$, in r random strings of length n generated by a Bernoulli process.

r	n	Indep. Words Approx.	Monte-Carlo Est.
2	18	0.0490	0.0386
4	197	0.0497	0.0477
6	467	0.0499	0.0490
8	727	0.0500	0.0493
10	958	0.0501	0.0495
2	11	0.0107	0.0088
4	131	0.0100	0.0096
6	346	0.0099	0.0098
8	569	0.0100	0.0099
10	774	0.0100	0.0100

Independent Words Model

- Time complexity: $O(\sigma^{\mathbf{k}})$
    - Where σ is the size of the input alphabet

- Observation: We can reduce the time complexity by grouping together strings that have the same probability of occurrence in random strings

Double Independence Model

- **Assumption 1**: Different *k*-substrings occur independently of each other in random strings
- **Assumption 2**: The self-overlap structure of a *k*-substring can be ignored when calculating the probability that it occurs in a random string

Double Independence Model

- Approach:
  1. Define the *composition* for strings of length *k* such that strings that share the same composition have the same probability of occurring in a random string
  2. Enumerate every composition for strings of length *k*.

Double Independence Model

- Resulting approximation:

$$P(\zeta) = 1 - \prod_{\gamma \in \mathcal{C}_k} (1 - P_\gamma{}^r)^{\Omega(\gamma)}$$

- Notation:

$\mathcal{C}_k$ – Set of all compositions for strings of length *k*

P_γ – Probability of occurrence for strings with composition γ

$\Omega(\gamma)$ – Number of strings with the composition γ

Double Independence Model: Bernoulli Process

- The *Bernoulli composition* of a string w is the multiset γ of characters in w.
  - Example: The composition of the string ACCATA is $\gamma = \{A, A, A, C, C, T\}$

Double Independence Model: Bernoulli Process

$$P(\zeta) = 1 - \prod_{\gamma \in \mathcal{C}_k} (1 - P_\gamma{}^r)^{\Omega(\gamma)}$$

- The probability P_γ can be computed easily
- Enumeration of the compositions in $\mathcal{C}_k$ can be accomplished with a simple recursive algorithm
- Number of strings with the composition γ given by multinomial equation.

Double Independence Model: Bernoulli Process

Table 1: Approximations to CSRS for $k = 6$, in r random strings of length n generated by a Bernoulli process.

r	n	DIM Approx.	IWM Approx.	Monte-Carlo
2	18	0.0491	0.0490	0.0386
4	197	0.0500	0.0497	0.0477
6	467	0.0508	0.0499	0.0490
8	727	0.0514	0.0500	0.0493
10	958	0.0519	0.0501	0.0495
2	11	0.0107	0.0107	0.0088
4	131	0.0101	0.0100	0.0096
6	346	0.0101	0.0099	0.0098
8	569	0.0103	0.0100	0.0099
10	774	0.0104	0.0100	0.0100

Double Independence Model: Bernoulli Process

- Time Complexity: $O(k^{\sigma-1})$

\+ Running time polynomial in k

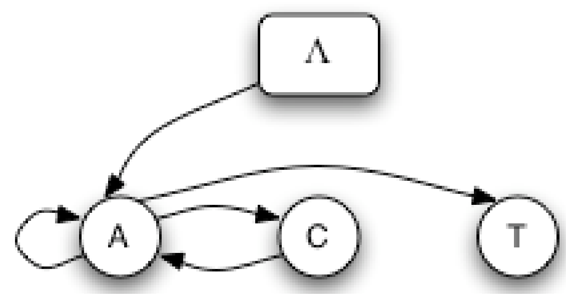
– Less accurate than the Independent Words Model

Double Independence Model: Markov Process

- The *1st-order Markov composition* of a string *w* is the multiset γ of Markov transitions between adjacent characters in *w* and between the start state Λ and the first character in *w*
  - Example: The 1st-order Markov composition of the string AACAT is $\gamma = \{(\Lambda \to A), (A \to A), (A \to C), (A \to T), (C \to A)\}$

Double Independence Model: Markov Process

- The 1st-order Markov composition of a string can also be represented as a directed multigraph.



  – Example: The 1st-order Markov composition graph of the string AACAT.

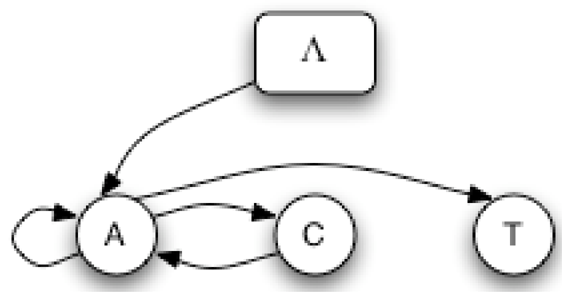
Double Independence Model: Markov Process

$$P(\zeta) = 1 - \prod_{\gamma \in \mathcal{C}_k} (1 - P_\gamma{}^r)^{\Omega(\gamma)}$$

- P_γ is easy to compute
- **Challenge 1**: Counting the number of strings that share the Markov composition γ
- **Challenge 2**: Enumerating all the compositions in $\mathcal{C}_k$

Double Independence Model: Markov Process

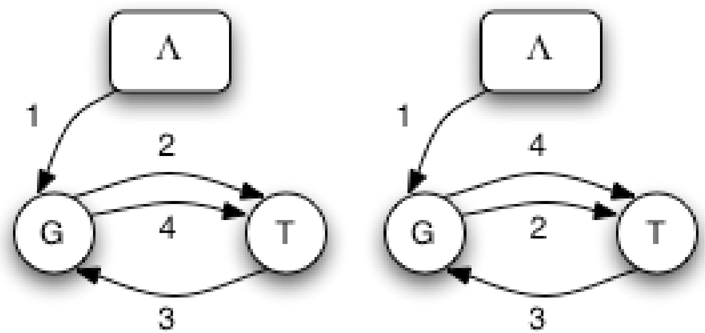
- To count the number of strings with the composition γ, we will count the number of Eulerian trails on the Markov composition graph for γ.



- Counting the number of Eulerian trails on a directed multigraph is done with the BEST theorem [van Aardenne-Ehrenfest and de Bruijn, 1951].

Double Independence Model: Markov Process

- **But**: some distinct Eulerian trails correspond to the same string,



Double Independence Model: Markov Process

- Result: The number of strings with the composition γ is defined by

$$\Omega(\gamma) = \frac{\lambda_\gamma}{\prod_{(u,v)\in V_\gamma{}^2} M(u,v)!}$$

- Notation:

λ_γ – The number of Eulerian trails on the Markov composition graph γ

$M(u,v)$ – The number of edges going from *u* to *v* in the Markov composition graph

Double Independence Model: Markov Process

Table 1: Approximations to CSRS for $k = 6$, in r random strings of length n generated by a 1st-order Markov process.

r	n	DIM	DIM w. Correction	Monte-Carlo
2	17	0.0497	0.0495	0.0388
4	175	0.0522	0.0493	0.0474
6	410	0.0549	0.0493	0.0486
8	633	0.0607	0.0494	0.0493
10	828	0.0674	0.0494	0.0493
2	10	0.0088	0.0088	0.0073
4	117	0.0105	0.0101	0.0098
6	304	0.0113	0.0099	0.0099
8	494	0.0128	0.0099	0.0099
10	666	0.0148	0.0098	0.0099

Generalizations

- The Models can also be modified to handle modifications to the original CSRS problem, including:
  - Searching for common substrings in a subset of the random strings
  - Allowing mismatches in the substring occurrences within the random strings

Future Work

- Providing theoretical bounds on the error introduced by the approximations
- Improving the quality of the approximation when the number of strings is low
- Developing new models with weaker assumptions

Acknowledgements

- Uri Keich
- Members of the McGill Centre for Bioinformatics

- Funding by
  - Andre Courtemanche Fellowship
  - NSERC