ERATO セミナ 2010 - No. 25

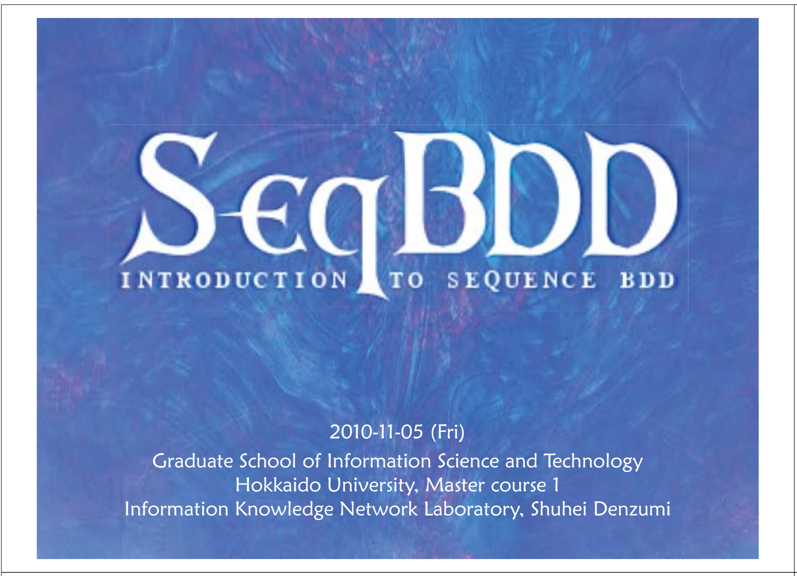
Seq BDD と既存手法との比較について

伝住 周平
北海道大学 情報科学研究科修士過程

2010/11/5

概要

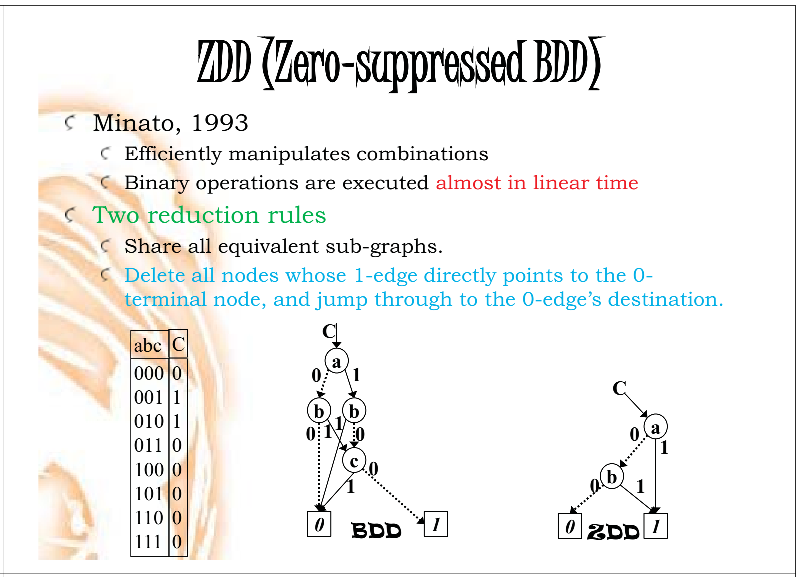
ZDD から派生したデータ構造として，文字列集合や系列（sequential）データを表現するための seqBDD が提案されている．本発表では seqBDD の概説を行う．特に，１つの文字列の全ての部分文字列を表現する手法について，既存の DAWG などのデータ構造との比較を行う．



SeqBDD

INTRODUCTION TO SEQUENCE BDD

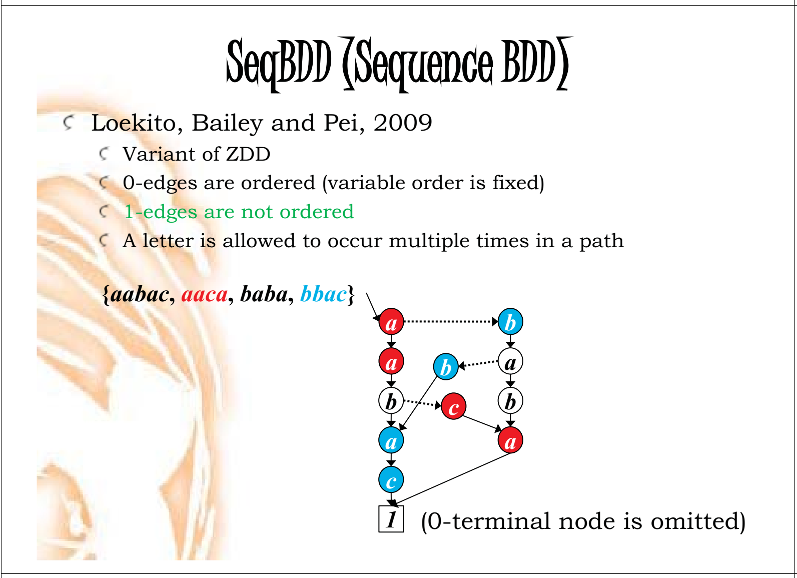
2010-11-05 (Fri)
Graduate School of Information Science and Technology
Hokkaido University, Master course 1
Information Knowledge Network Laboratory, Shuhei Denzumi



ZDD (Zero-suppressed BDD)

- Minato, 1993
  - Efficiently manipulates combinations
  - Binary operations are executed almost in linear time
- Two reduction rules
  - Share all equivalent sub-graphs.
  - Delete all nodes whose 1-edge directly points to the 0-terminal node, and jump through to the 0-edge's destination.

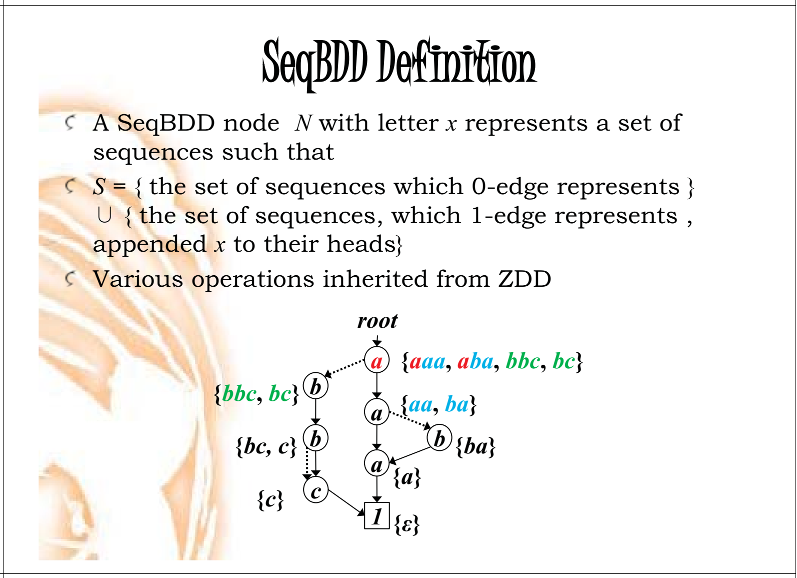
abc	C
000	0
001	1
010	1
011	0
100	0
101	0
110	0
111	0



SeqBDD (Sequence BDD)

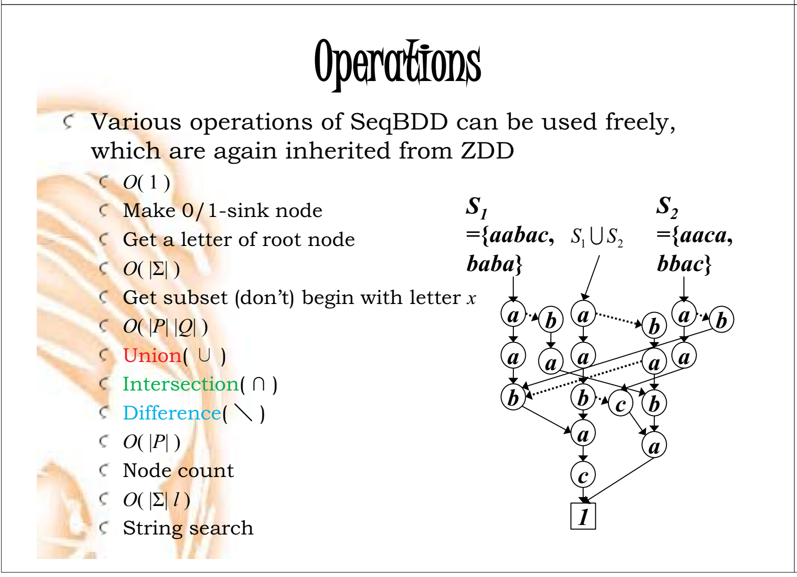
- Loekito, Bailey and Pei, 2009
  - Variant of ZDD
  - 0-edges are ordered (variable order is fixed)
  - 1-edges are not ordered
  - A letter is allowed to occur multiple times in a path

(0-terminal node is omitted)



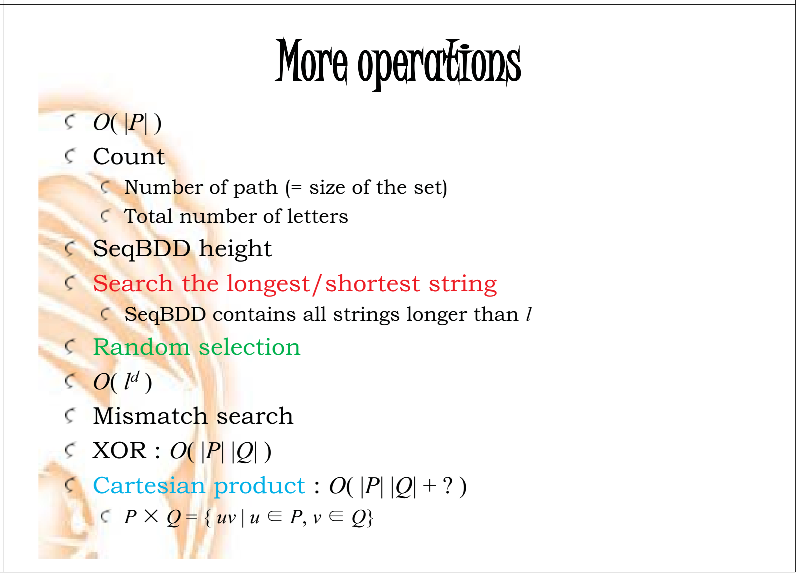
SeqBDD Definition

- A SeqBDD node N with letter x represents a set of sequences such that
- S = { the set of sequences which 0-edge represents } ∪ { the set of sequences, which 1-edge represents , appended x to their heads}
- Various operations inherited from ZDD



Operations

- Various operations of SeqBDD can be used freely, which are again inherited from ZDD
  - $O(1)$
  - Make 0/1-sink node
  - Get a letter of root node
  - $O(|\Sigma|)$
  - Get subset (don't) begin with letter x
  - $O(|P|\,|Q|)$
  - Union(∪)
  - Intersection(∩)
  - Difference(＼)
  - $O(|P|)$
  - Node count
  - $O(|\Sigma|\,l)$
  - String search



More operations

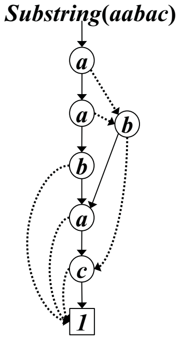
- $O(|P|)$
- Count
  - Number of path (= size of the set)
  - Total number of letters
- SeqBDD height
- Search the longest/shortest string
  - SeqBDD contains all strings longer than l
- Random selection
- $O(l^d)$
- Mismatch search
- XOR : $O(|P|\,|Q|)$
- Cartesian product : $O(|P|\,|Q| + ?)$
  - $P \times Q = \{ uv \mid u \in P, v \in Q \}$

Development

- SuffixDD
  - Store the all substrings of a text
  - Possible input
    - Set of texts (Generalized SuffixDD)
    - SeqBDD
- SubseqDD
  - Store the all subsequences of a text
- SeqBDD vector
  - Add, subtract, greater than, less than
  - Weighted SeqBDD (Loekito et al.)
- Search
  - Position, ID
  - Substring, subsequence
- Wild card
  - Search

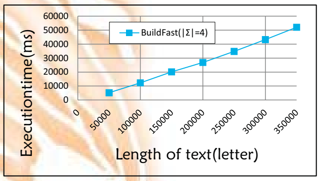
SuffixDD

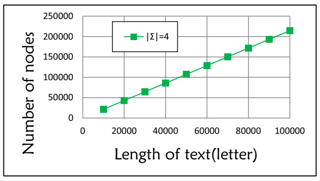
- Suffix Decision Diagram
  - Substring index on SeqBDD
- Represents the set of all substrings of a text
- The number of nodes
  - n+1 in the best case
  - $3n-2$ in the worst case
- The number of edges is twice as mush as nodes
- Space complexity is linear
- Naïve construction in $O(n^3)$ time
- A faster construction algorithm in $O(n^2)$ time
- Can prove $O(n)$ with operation cache?



Time & Space

- Measured execution time of efficient algorithm and the size of SuffixDD
- Input : random string





- Computation time of fast algorithm looks $O(n)$
- In theory, $O(n^2)$
- The size of SuffixDD is almost twice to the text length
- SuffixDD with larger alphabet size are slightly smaller

Experiment

File	File size (B)	SuffixDD size	#Substrings	#letters	Time (ms)
paper1	53,161	102,025	1.41×10^9	2.50×10^{13}	25,323
paper2	82,199	157,398	3.38×10^9	9.26×10^{13}	43,391
paper3	46,526	89,941	1.08×10^9	1.68×10^{13}	22,344
paper4	13,286	26,078	88,196,012	3.91×10^{11}	4,443
paper5	11,954	23,243	71,392,689	2.85×10^{11}	4,297
paper6	38,105	73,989	725,674,256	9.22×10^{12}	16,261
Sum	245,231	472,674	6.76×10^9	1.44×10^{14}	–
Union	–	470,534	6.76×10^9	1.44×10^{14}	2,079
Intersection	–	2,397	5,280	24,409	521

```
Eshell V5.7.3  (abort with ^G)
1> seqbdd:set_table().                        ==> ok
2> S1 = sdd:suffixdd(seq:read("paper1")).   ==>  4379855
3> S2 = sdd:suffixdd(seq:read("paper2")).   ==>  11555546
4> S3 = sdd:suffixdd(seq:read("paper3")).   ==>  15431702
5> S4 = sdd:suffixdd(seq:read("paper4")).   ==>  16299568
6> S5 = sdd:suffixdd(seq:read("paper5")).   ==>  17134036
7> S6 = sdd:suffixdd(seq:read("paper6")).   ==>  20018804
8> I = sdd:intersect(sdd:intersect(S1,S2),S3).  ==>  20042241
9> U = sdd:union(sdd:union(S4,S5),S6).      ==>  20089690
10> D = sdd:difference(I, U).               ==>  20094751
11> sdd:longest(D).  ==>
   "\n.sp2\n.ce4\nDepartment of Computer Science\nThe University
   of Calgary\n2500 University Drive NW\nCalgary, Canada T2N 1N4
   \n.sp2\n."
```

Latest

- Super maximal strings
  - Maximal strings
- Hamming (Edit) distance
  - With window
  - Using all alphabet/wild card
- Division and remainder : $O(\,|P|\,|Q| + ?)$
  - Not implemented
  - $P \diagdown_{\mathrm{E}} Q = \{v \mid \exists u \in P, uv \in Q\}$, $P \diagdown_{\mathrm{A}} Q = \{v \mid \forall u \in P, uv \in Q\}$
  - $Q \diagup_{\mathrm{E}} P = \{v \mid \exists w \in P, vw \in Q\}$, $Q \diagup_{\mathrm{A}} P = \{v \mid \forall w \in P, vw \in Q\}$
  - $P\ {}_{\mathrm{L}}\%_{\mathrm{E}}\ Q = \{uv \mid u \notin P, uv \in Q\}$, $P\ {}_{\mathrm{L}}\%_{\mathrm{A}}\ Q = Q - (P \times (P \diagdown_{\mathrm{A}} Q))$
  - $Q\ {}_{\mathrm{R}}\%_{\mathrm{E}}\ P = \{vw \mid w \notin P, vw \in Q\}$, $Q\ {}_{\mathrm{R}}\%_{\mathrm{A}}\ P = Q - ((Q \diagup_{\mathrm{A}} P) \times P)$
- Factor : $O(?)$
  - $\{uw \mid \exists v \in P, uvw \in Q\}$, $\{uw \mid \forall v \in P, uvw \in Q\}$
  - $\{uvw \mid v \notin P, uvw \in Q\}$