IP Address Lookup and Packet Classification Algorithms

Zhen Xu, Jeff Nie, Xuehong Sun, and Yiqiang Q. Zhao

School of Mathematics and Statistics, Carleton University

Outline

1. Background

2. Two IP Address Lookup Algorithms

   a. **Fast IP Address Lookup Using Index Table**

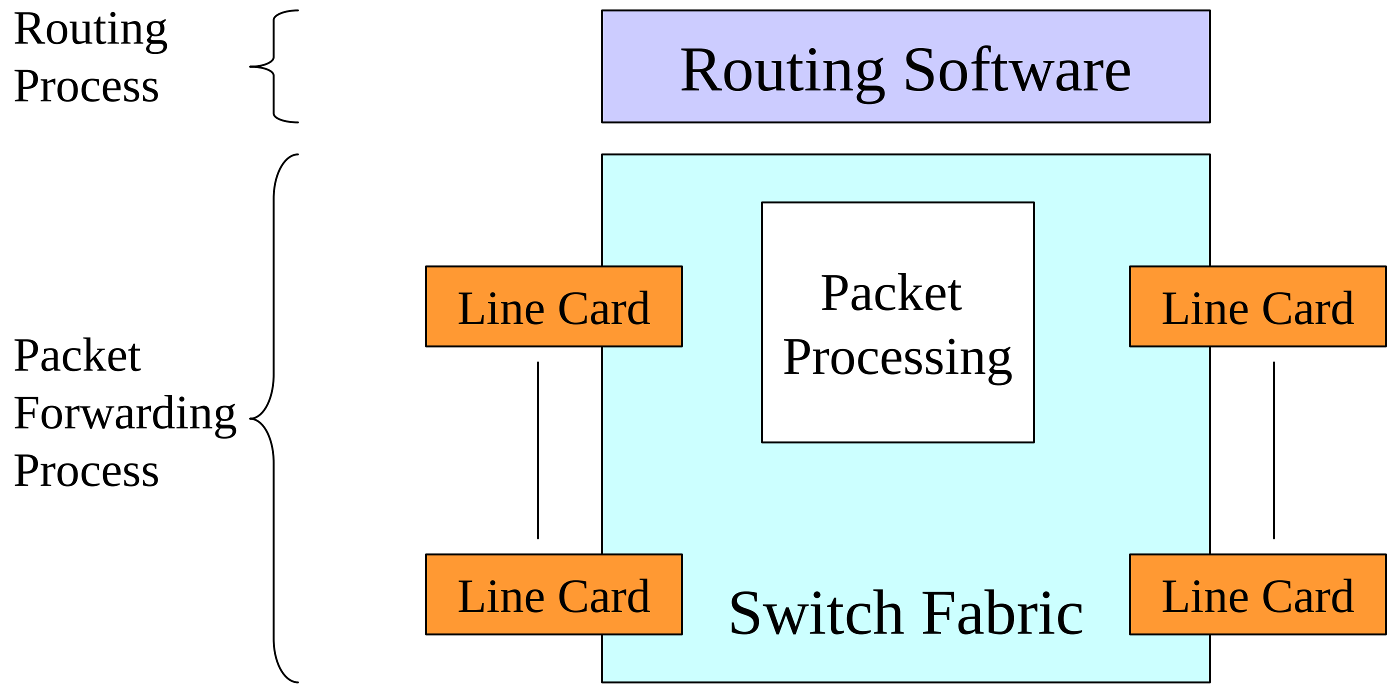
   b. **Fast IP Address Lookup Based on Comb Extraction Scheme**

3. One Packet Classification Algorithm

   a. **Packet Classification Using Independent Sets**

Background of IP Address Lookup and Packet Classification

Architecture of Packet-by-packet routers:



An Example of Forwarding Table:

ENTRY NUMBER	DESTINATION PREFIX	NEXT-HOP
1	192.2.0.0/22	100.1.2.1
2	200.11.0.0/22	120.3.5.3
3	192.2.2.0/24	120.3.5.3

An Example of Classifier:

Rule	Network-layer Dest. prefix	Network-layer Sourc. prefix	Trans-layer Dest.	Trans-layer protocol	Action
R1	152.163.190.69/32	152.163.80.11/32	*	*	Deny
R2	152.168.3.0/24	152.163.20.157/32	Eq http	Udp	Deny
R3	152.168.3.0/24	152.163.20.157/32	Range 20-21	Udp	Permit
R4	152.163.198.4/32	152.163.0.0/16	Gt 1023	Tcp	Deny
R5	*	*	*	*	Permit

Classification is a Generalization of Lookup

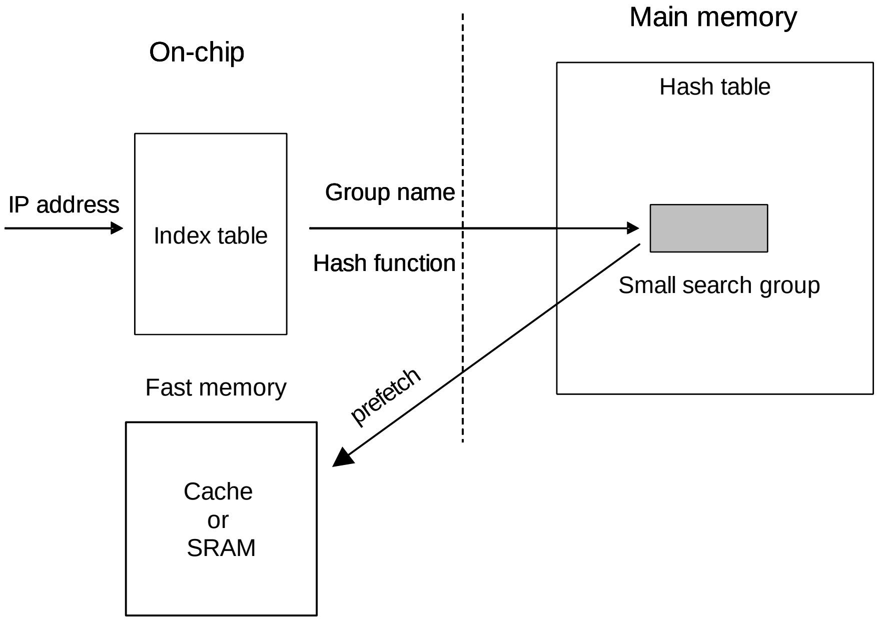
- ✍ Classifier = Forwarding Table
- ✍ One dimension (destination address)
- ✍ Rule = Forwarding Table Entry
- ✍ Action = (Next-hop-address, out port)
- ✍ Priority = Prefix Length

IP Address Lookup Algorithms

Fast IP Address Lookup Using Index Table

Proposed by Jeff Nie

The Idea of the Proposed Algorithm

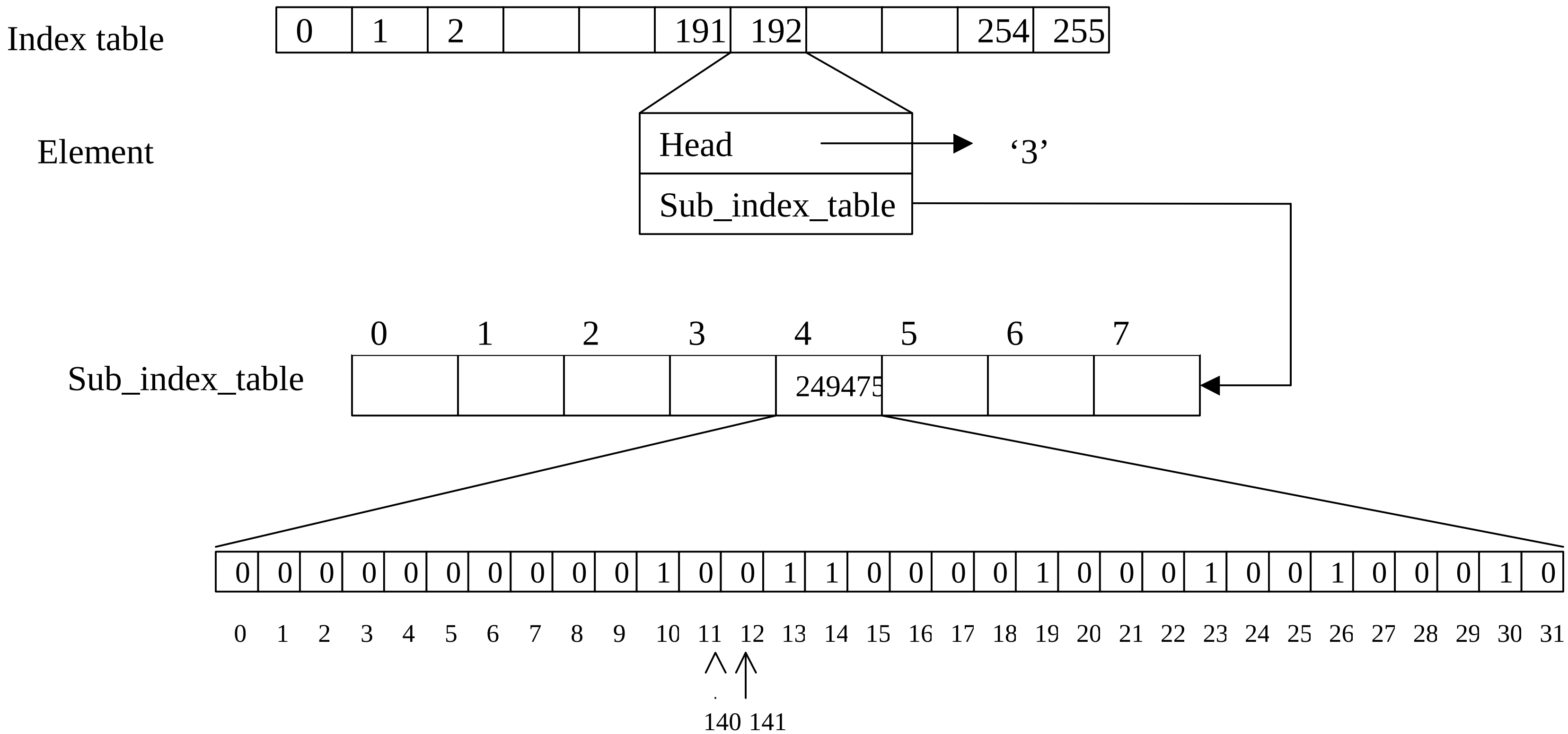


- The index table stores the length information of prefixes.
- All prefix entries are stored in a hash table in which hash keys are bit strings of different length.
- Prefixes with the same key are grouped into a small table,called small search group.
- The number of prefixes in a small search group can be controlled.
- The small search group can be prefetched into fast on-chip memories on a general-purpose CPU to improve the lookup rate.

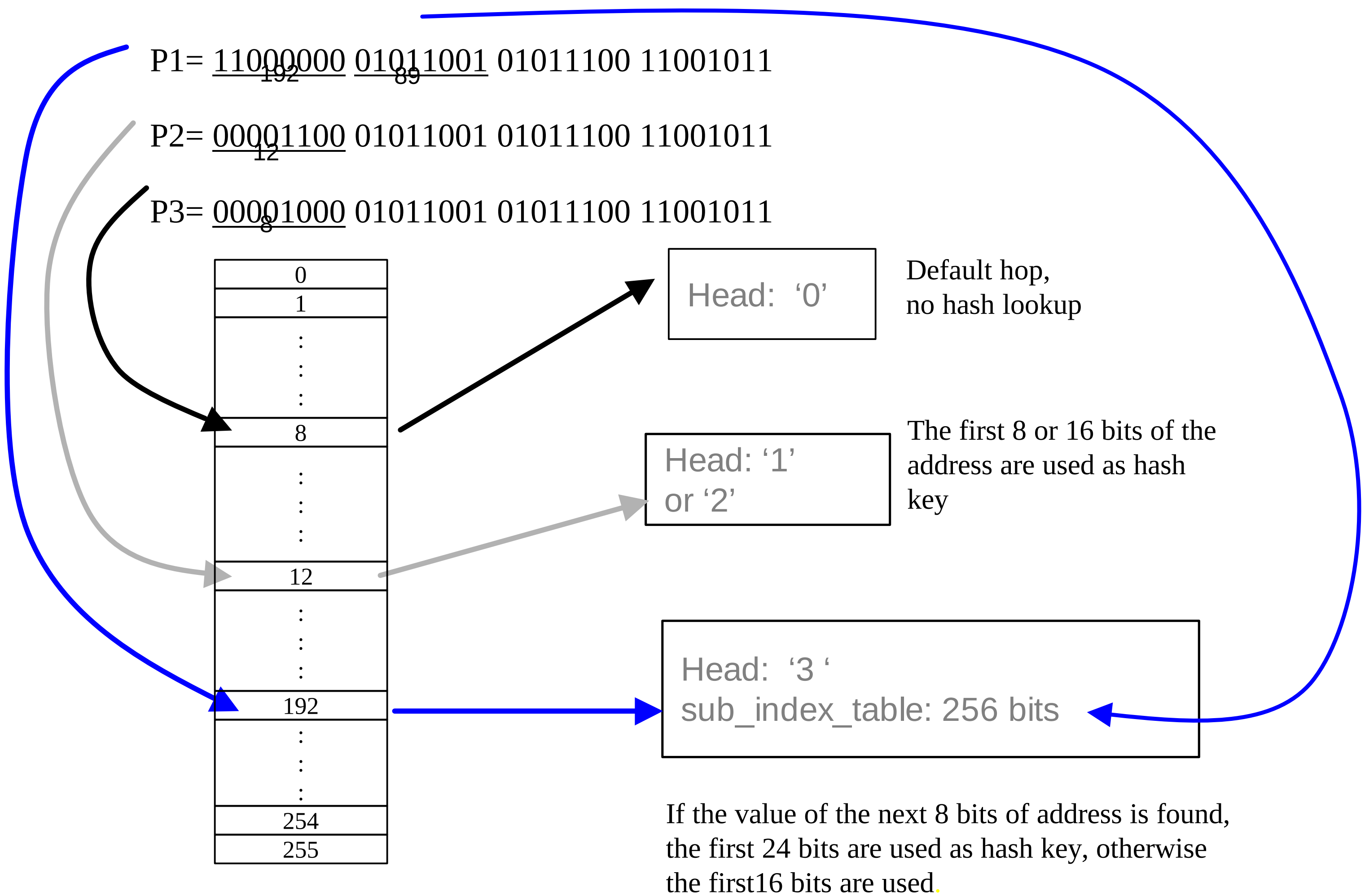
Index Table

- An array with total 256 elements.
- Each element has two items: head and sub_index_table.
- Head has four kinds of values: 0, 1, 2 or 3, which represents that the first 0, 8, 16 or 24 bits of an IP address are extracted as hash key, respectively.
- A sub_index_table stores the state of the next 8 bits of a prefix.
- The decimal value of the next 8 bits of an IP address is used as index in index table
- If the head is equal to 3, the next 8 bits of the IP address are extracted to check its state in the sub_index_table. If its state is 1, the head is 3, otherwise the head is 2.

Index Table (cont.)



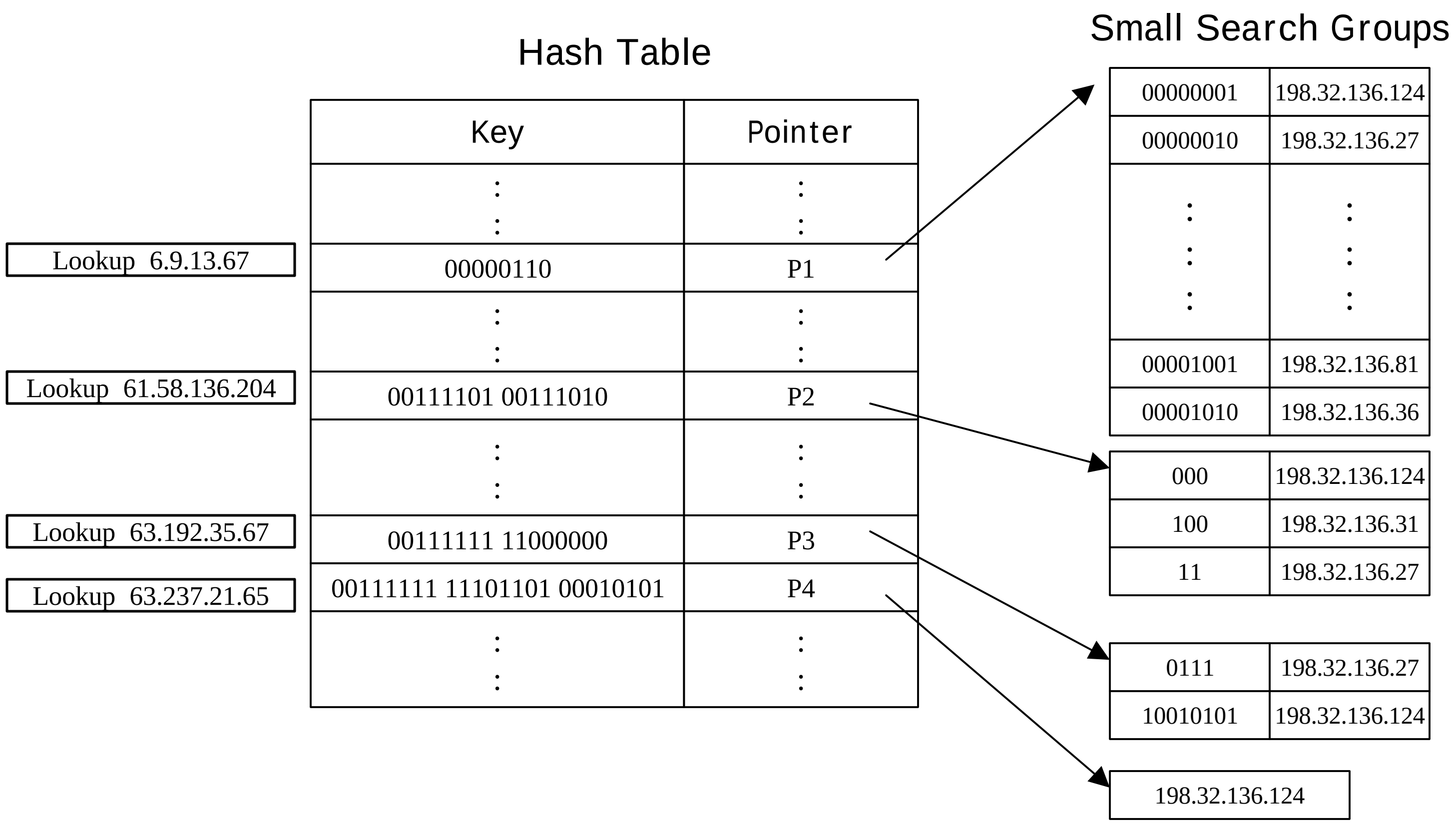
An Example of Searching the Index Table



Hash Table

- ✍ Hash table is organized by string bit with different lengths
- ✍ Each element has two items: a key and a pointer of information or of a small search group.
- ✍ A small amount of prefixes need to be extended to a certain length.

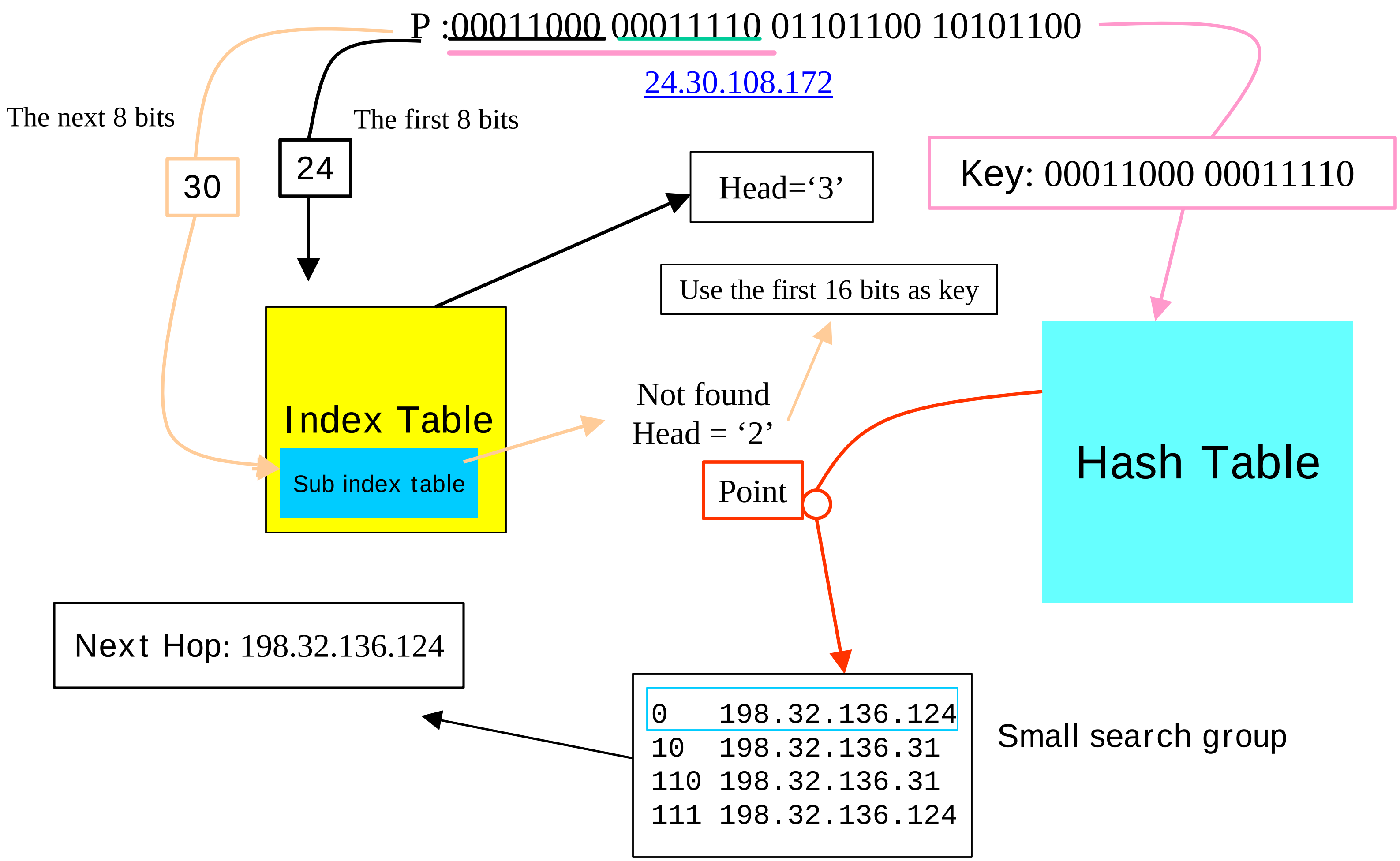
An Example of the Hash Table



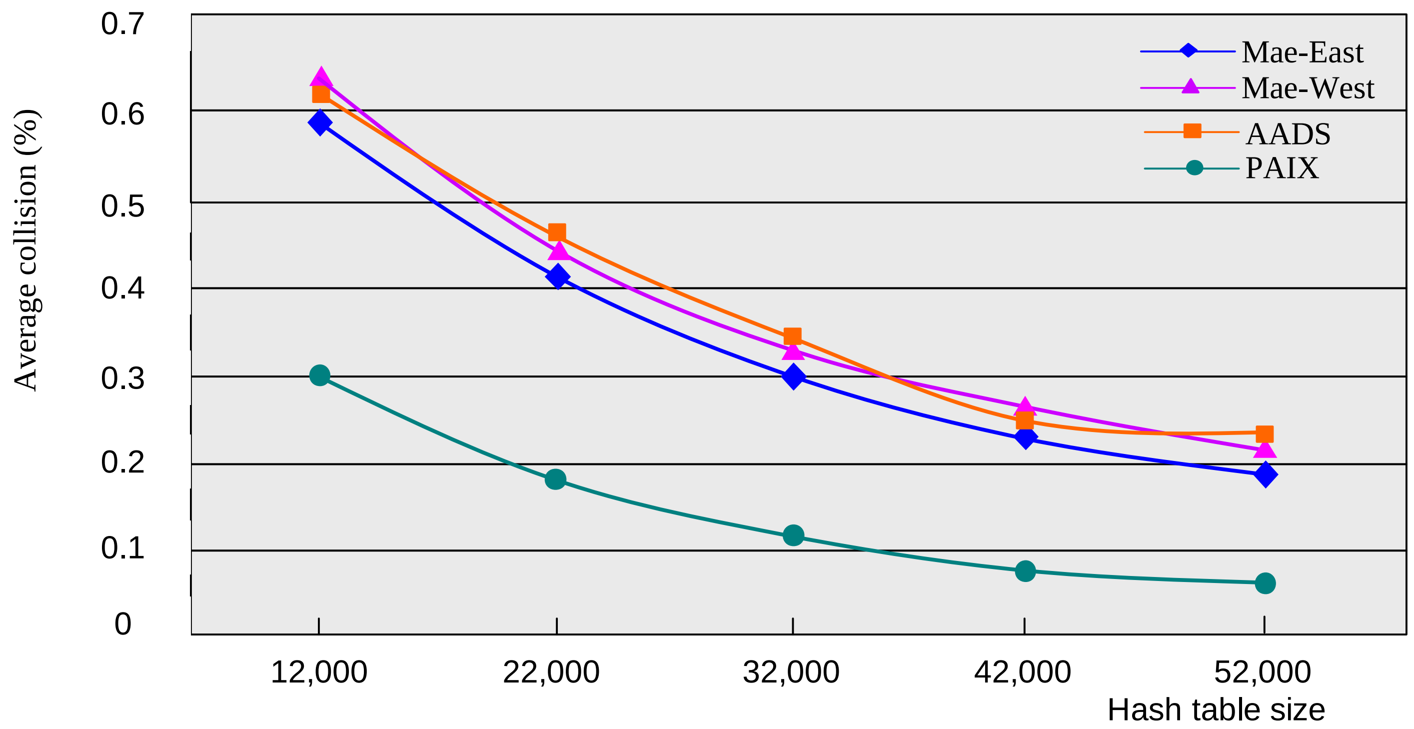
Small Search Groups

- An array of at most 16 items
- Each item is a structure consisting of two 32-bit words
  - The first word stands for the part of a prefix, extension flag and the prefix length.
  - The second word stands for the next hop.
- All items in small search group are ordered by decreasing length.
- All items are arranged on the consecutive memories so that they can be retrieved into the L1 cache on CPU with one memory access.
- The size of the small search group can be changed to fit it into different capacity RAM.

Search

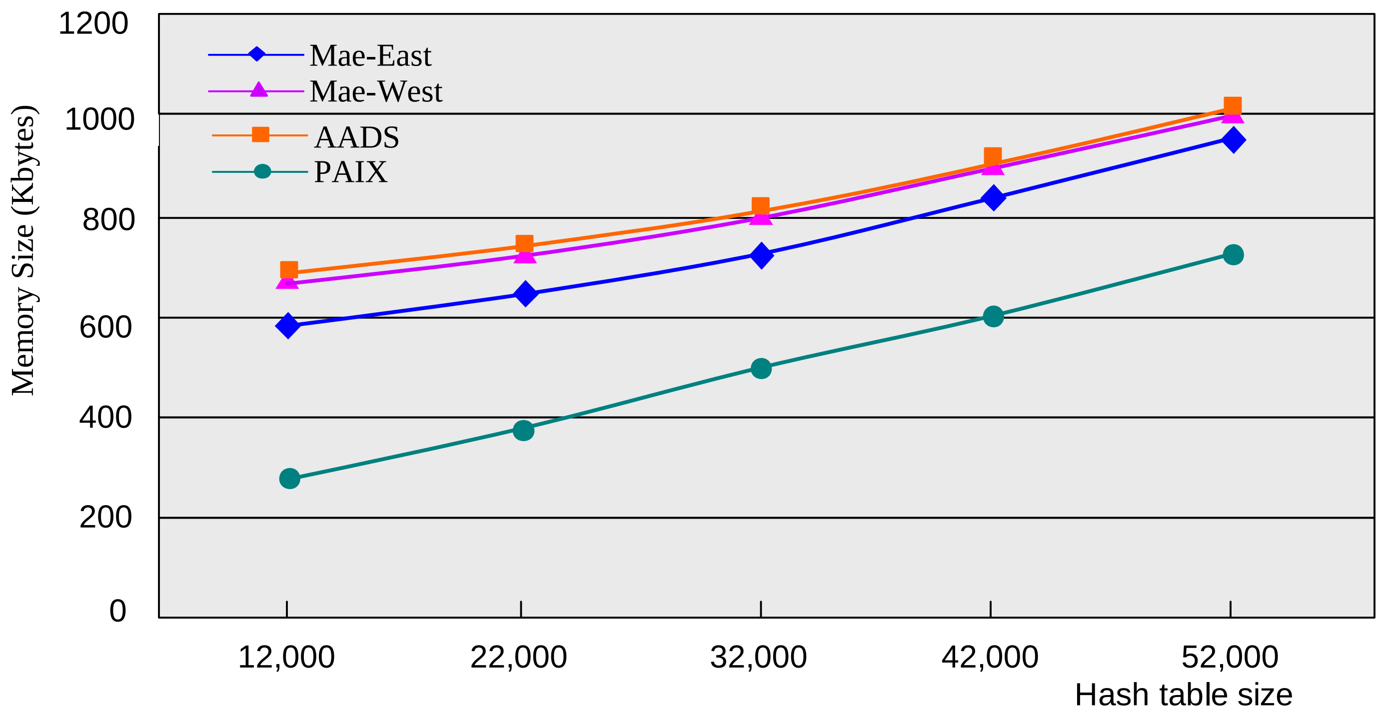

P :00011000 00011110 01101100 10101100
24.30.108.172
The next 8 bits
The first 8 bits
30
24
Head='3'
Key: 00011000 00011110
Use the first 16 bits as key
Index Table
Sub index table
Not found
Head = '2'
Point
Hash Table
Next Hop: 198.32.136.124
0 198.32.136.124
10 198.32.136.31
110 198.32.136.31
111 198.32.136.124
Small search group

Hash Table Size



Average collisions versus the hash table size

Memory Usage



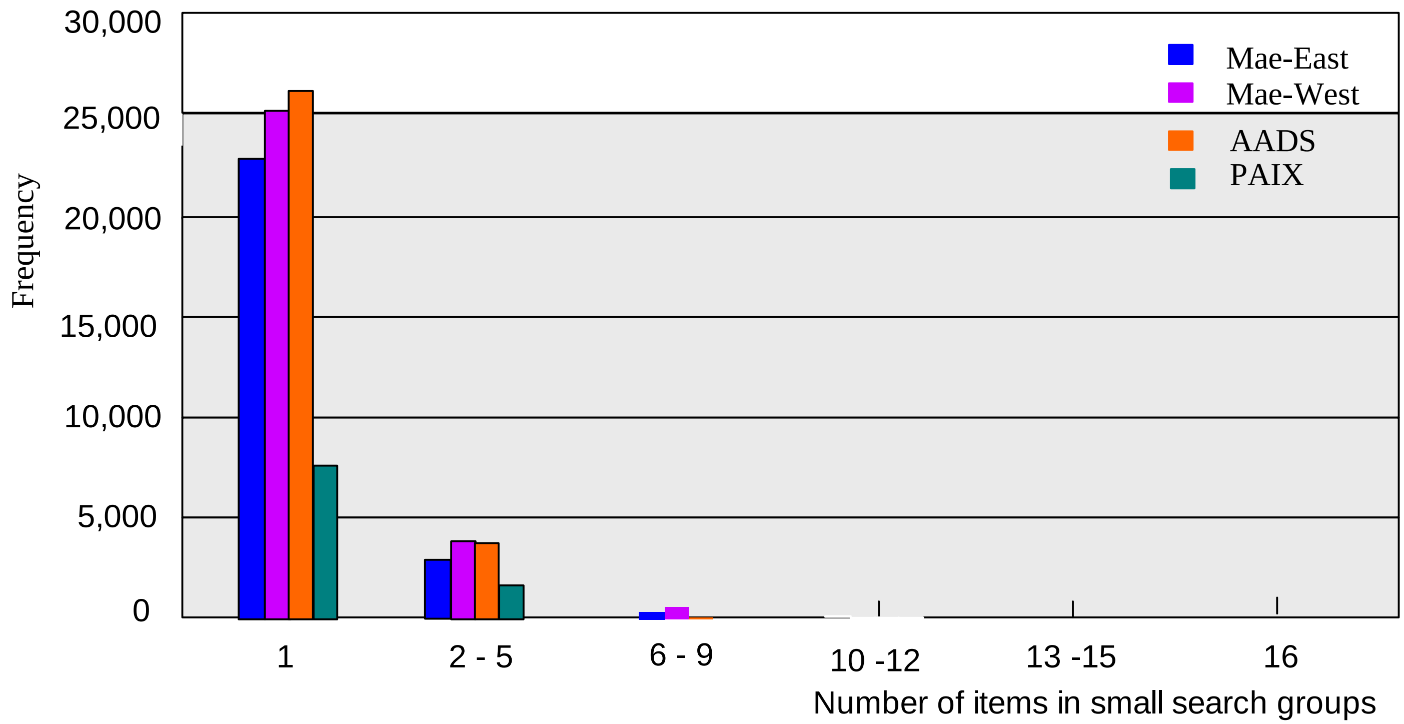
Memory usage versus the hash table size.

Hash lookup

The hash lookup results on the simulation

Times of hash lookup	Mae-East		Mae-West		AADS		PAIX	
	Number	%	Number	%	Number	%	Number	%
1	14696	73.34	15489	70.52	16089	71.17	8273	92.97
2	4514	22.53	5363	24.42	5286	23.38	591	6.64
3	739	3.69	985	4.48	1084	4.79	33	0.37
4	85	0.42	117	0.53	139	0.61	2	0.02
5	3	0.01	8	0.04	10	0.04	0	0
6	0	0	1	0.005	0	0	0	0
Average	1.31		1.35		1.35		1.07	
Worst case	5		6		5		4	

Frequency histogram of the number of items in the small search groups



Conclusions

- Using the index table reduces the searching range size from the whole database to a small search group.
- On the average, only one main memory access is needed.
- Index table is small enough to fit into primary cache.
- The small search group is prefetched into the primary cache on general purpose CPU to improve the lookup speed.
- The size of the small search group can be controlled so that the whole group can be retrieved into the L1 cache on CPU with one memory access.
- The update of a prefix only involves a few memory locations being modified.
- Compared with other algorithms, the algorithm provides some advantages.

IP Address Lookup Based on Comb Extraction Scheme

Proposed by Zhen Xu

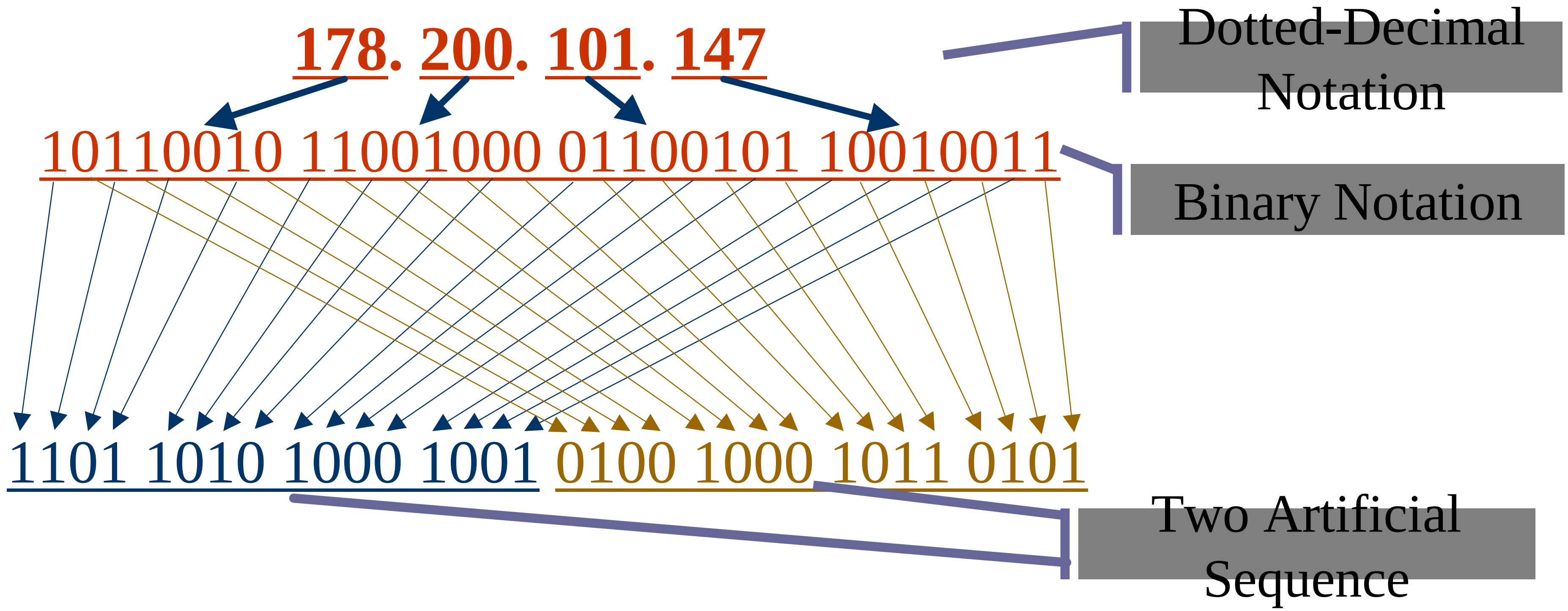
General Idea

1. One forwarding table is decomposed into two balanced smaller sub-forwarding tables.

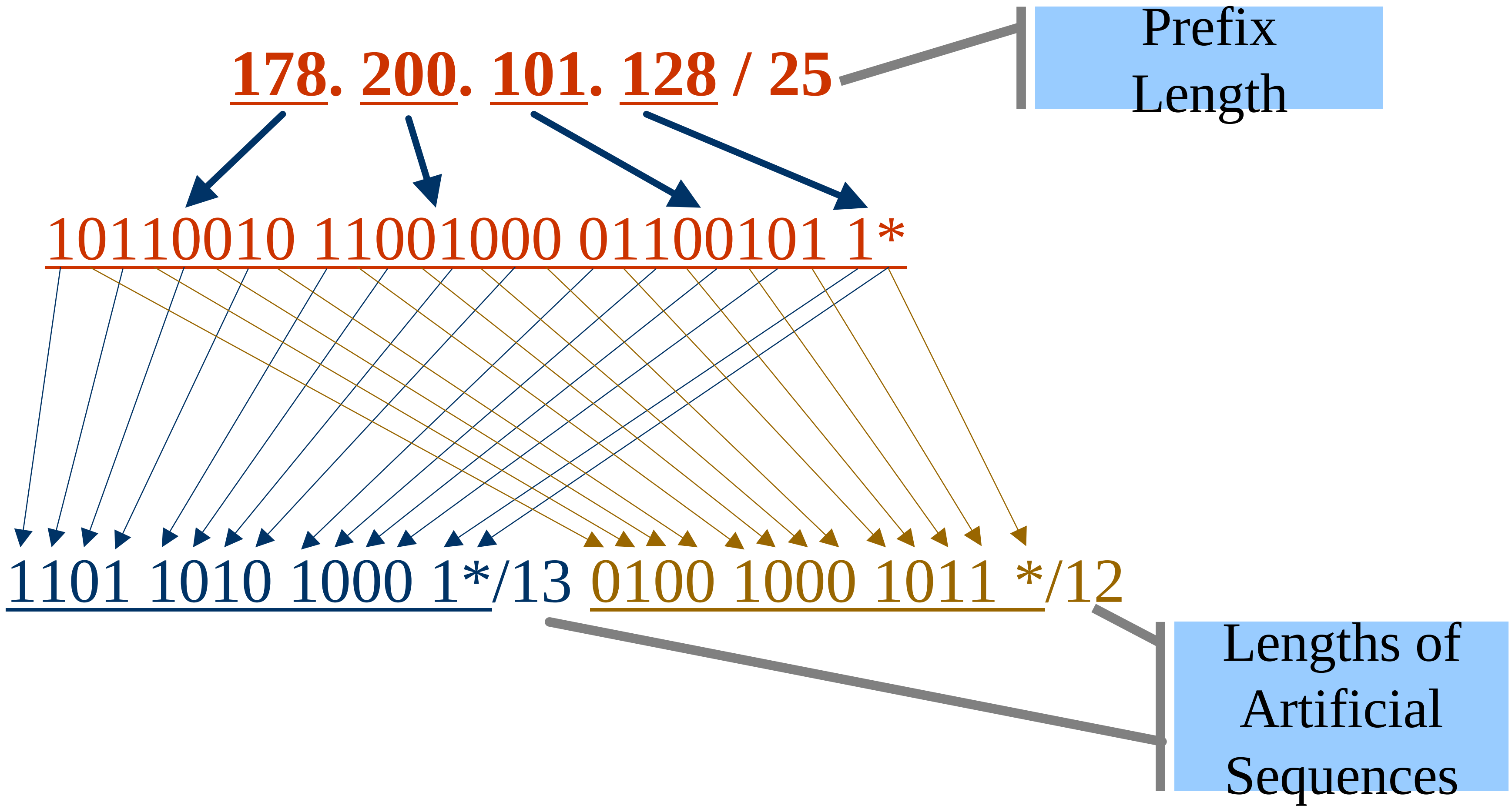
2. An IP lookup can be converted into two parallel small sub-lookups

3. After comparing the information attached by matching sub-prefixes from both sub-lookups, the output of an incoming packet can be determined.

An Example of CES used in an IP Address:



An Example of CES used in an IP Address Prefix:



Splitting Forwarding Table:

No.	Des. Prefix	Output Port
1	**001***	1
2	**010***	2
3	**01000***	2
4	**101***	2
5	**0101***	3
6	**111***	4

No.	Sub-Des. Prefix	Forwarding Information
1	00*	2(2), 3(5)
2	000*	2(3)
3	01*	1(1)
4	11*	2(4), 4(6)

No.	Sub-Des. Prefix	Forwarding Information
1	0*	1(1), 2(4)
2	1*	2(2), 4(6)
3	10*	2(3)
4	11*	3(5)

Structure of each sub-entry:

1. The collection of **forwarding unit** *a(b)*, which implies that the original prefix of this sub-prefix is forwarded to port *a*, and the associated index is *b*. 20-bit long sequence is enough to represent a forwarding unit.

2. a N-bit ***port indicator vector*** is associated with every sub-entry. A bit is set in the bit vector if and only if the port occurs in its forwarding information. Usually the width of it is no more than 128.

Features of the Sub-tables:

1. CES makes the entries of two sub-tables well distributed.

   We define **Pseudo-Hamming Distance** (PHD) between two prefixes to determine the distance between two prefixes. *MPHD* is the mean of PHD of any two different prefixes in one table.

2. CES also balances the sub-prefix lengths in the two sub-tables.

   Mean Prefix Length (MPL)

3. CES makes the forwarding units well distributed in each sub-table.

   (a). In each sub-table, the Basic Load of Forwarding Information (BLFI) of the i^{th} sub-entry: BLFI_i ? NP_i,

   where NP_i is the total number of forwarding units of the i^{th} sub-entry.

(b). In each sub-table, the Mean Load of Forwarding Information (MLFI) of all sub-entries: $MLFI = \frac{1}{K}\sum_{i=1}^{K} BLFI_i$,

where K is the total number of sub-entries in this sub-table.

(c). In each sub-table, the Standard Deviation of Forwarding Information (SDFI):

$$SDFI = \sqrt{\frac{1}{K}\sum_{i=1}^{K} (BLFI_i - MLFI)^2}.$$

The smaller the SDFI, the better it is.

4. CES balances the comparison cost.

comparison cost factor (CCF) is used to judge whether the comparison load of those matching sub-prefixes in two sub-tables for an address lookup next is heavy or not. CCF is a statistical value from experiments, by counting the pairs really need to compare.

	Entries		Sub-entries	MPHD	MPL	Max(BLFI)	MLFI	SDFI	CCF	Storage Cost (in Byte)
Mae-east	47206	Sub-table 1	4026	55.22	11.18	93	11.73	13.41	8	186.06K
		Sub-table 2	5341	56.56	11.18	86	8.84	9.71		209.15K
Mae-west	77002	Sub-table 1	5703	56.47	11.22	100	13.05	15.49	8	270.81K
		Sub-table 2	6989	57.84	11.22	78	11.02	12.57		241.95K
Aads	63980	Sub-table 1	5689	56.80	11.35	110	11.25	14.28	8	245.14K
		Sub-table 2	6735	57.45	11.35	89	9.50	10.84		261.44K
Paix	22116	Sub-table 1	4077	54.59	11.15	40	5.42	5.35	7	117.65K
		Sub-table 2	4704	55.67	11.15	28	4.70	4.23		127.48K

Table 1: Performance of sub-tables by using the CES

	Entries		Sub-entries	MPHD	MPL	Max(BLFI)	MLFI	SDFI
Mae-east	47206	Sub-table 1	6939	59.85	16.00	280	6.80	13.63
		Sub-table 2	1349	43.24	6.47	2735	34.99	93.62
Mae-west	77002	Sub-table 1	10794	23.32	16.00	253	7.13	15.33
		Sub-table 2	1692	51.63	4.79	5939	45.50	164.45
Aads	63980	Sub-table 1	8314	61.54	16.00	465	7.69	16.65
		Sub-table 2	3540	49.03	9.94	3385	18.07	73.38
Paix	22116	Sub-table 1	4540	59.68	16.00	128	4.87	7.98
		Sub-table 2	1238	48.85	5.03	1406	17.86	49.95

Table 2: Performance of sub-tables by such a splitting rule: **extracting the higher 16 bits to form sub-table 1 and extracting the lower 16 bits to form sub-table 2**

Comparison

analyzing the matching sub-prefixes from two sub-tables common matching prefix. It can be implemented in an ASIC.

1. decide whether further comparing is necessary

If $?_1$ and $?_2$ are two the final matching sub-prefixes in the two-tables for an address, then they should satisfy the following:

(1) $|?_2|$ only can be equal to $|?_1|$ or $|?_1|-1$;

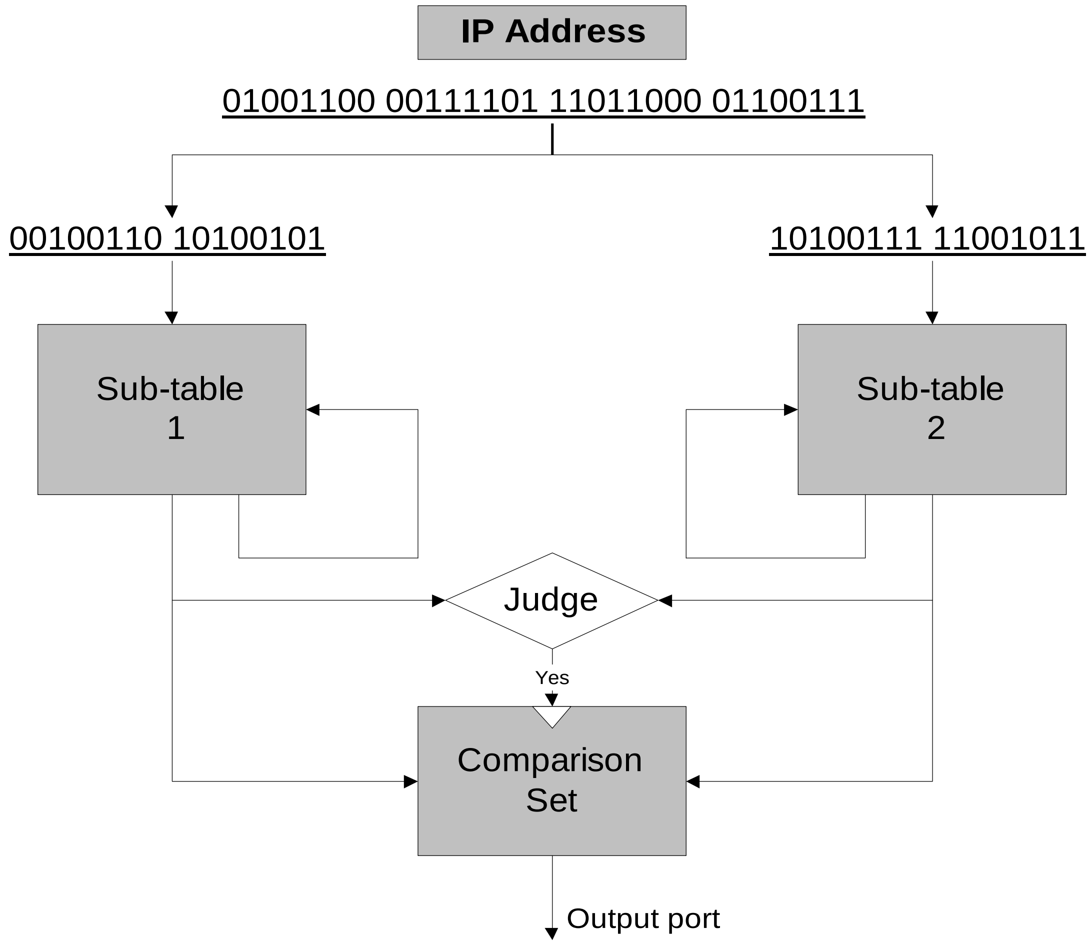
(2) In the two corresponding port indicator vectors,

$?i, i ? N, \mathrm{PIV}_i^1 ? \mathrm{PIV}_i^2 ? 1$ (PIV is the port indicator vector).

2. compare the forwarding units

	Entries	Average delay (ns)	Delay(80% of comparisons) (ns)
Mae-east	47206	2.24	<4.59
Mae-west	77002	3.36	<7.87
Aads	63980	1.96	<4.72
Paix	22116	0.56	<1.21

Search:


IP Address
01001100 00111101 11011000 01100111
00100110 10100101
10100111 11001011
Sub-table 1
Sub-table 2
Judge
Yes
Comparison Set
Output port

Performance:

1. The load of two sub-lookup systems keep in balance.
2. Overcome the problem of destination prefixes distribution dependent.
3. The heights of each extended LC trie is reduced.
4. Searching and comparison can work at the same time.
5. It is scalable for IPv6.
6. It can be extended to dynamic bits selection algorithms.
7. Since a forwarding table can be regarded as a 1-dimensional classifier, CES also can be used in packet classification.

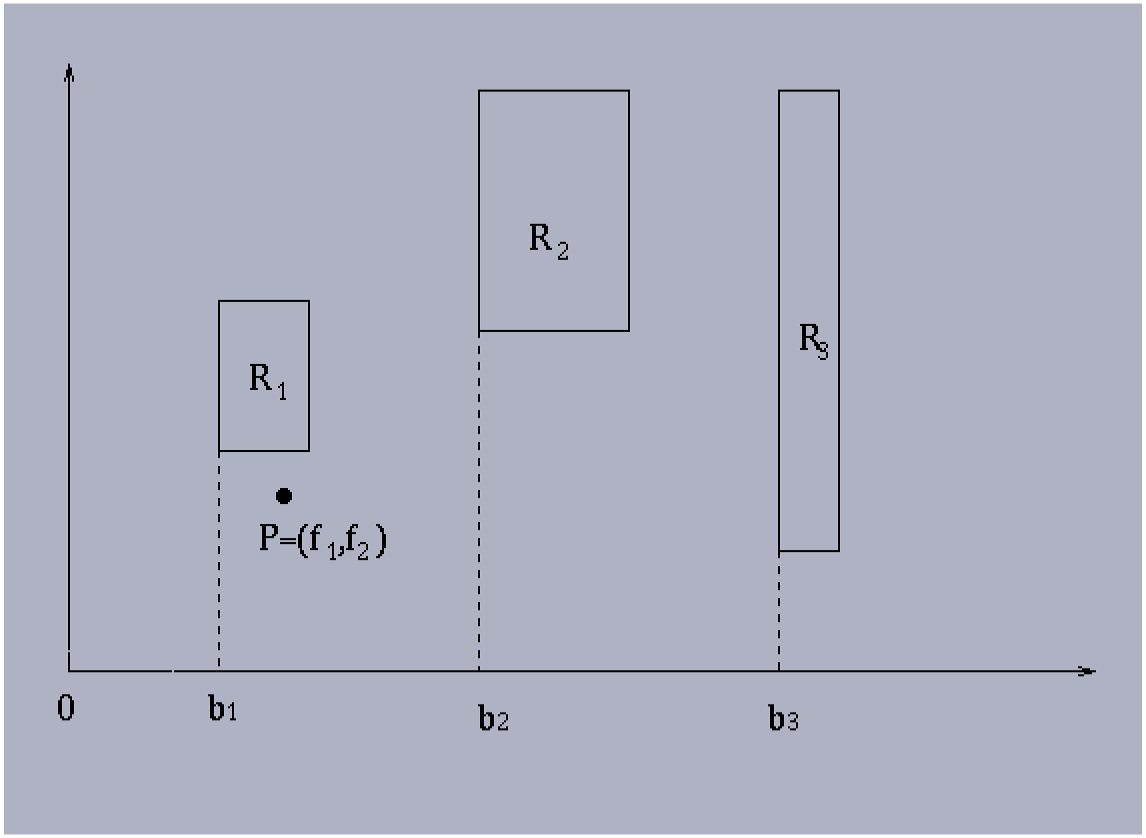
Packet Classification Using Independent Sets

Proposed by Xuehong Sun

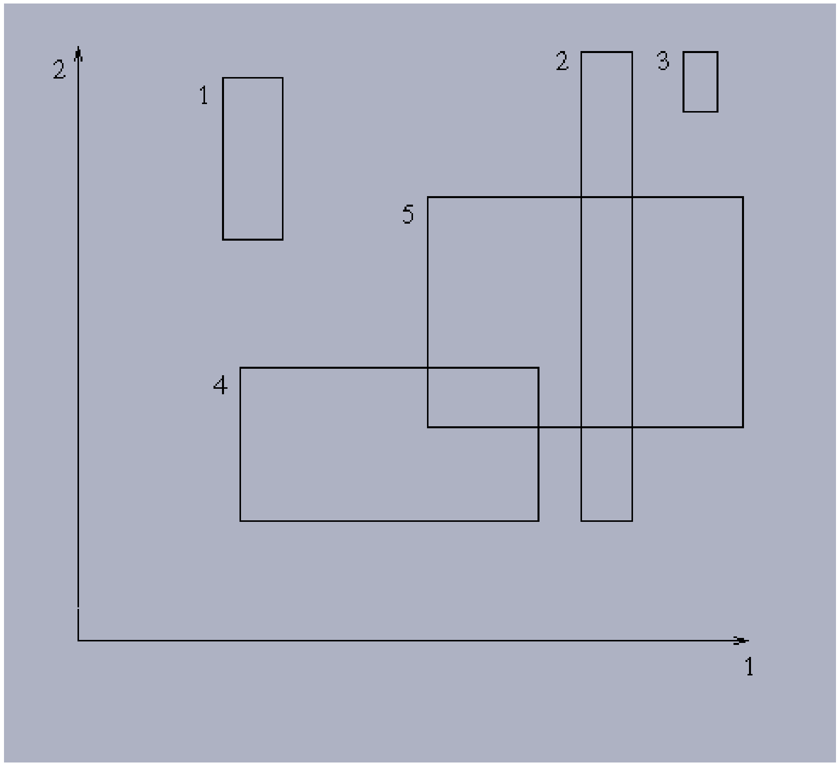
General idea

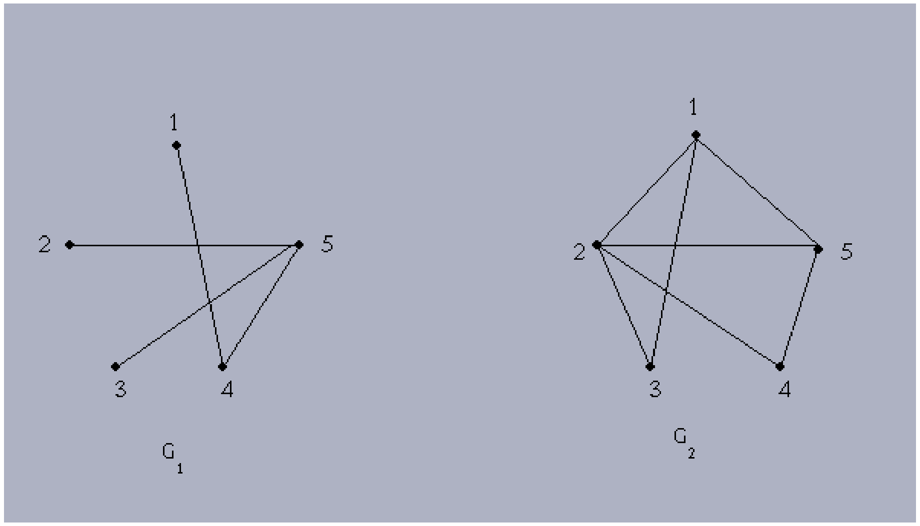
- a multiple-field key forms a rule.
- The relation between any two rules is not equal:
  - Some of them are “easy” to distinguish (independent); others not.
- All the rules that are easy to distinguish are grouped.
- All the rules are partitioned into these groups.
- A data structure is proposed to make the search easy.

An example of an independent set

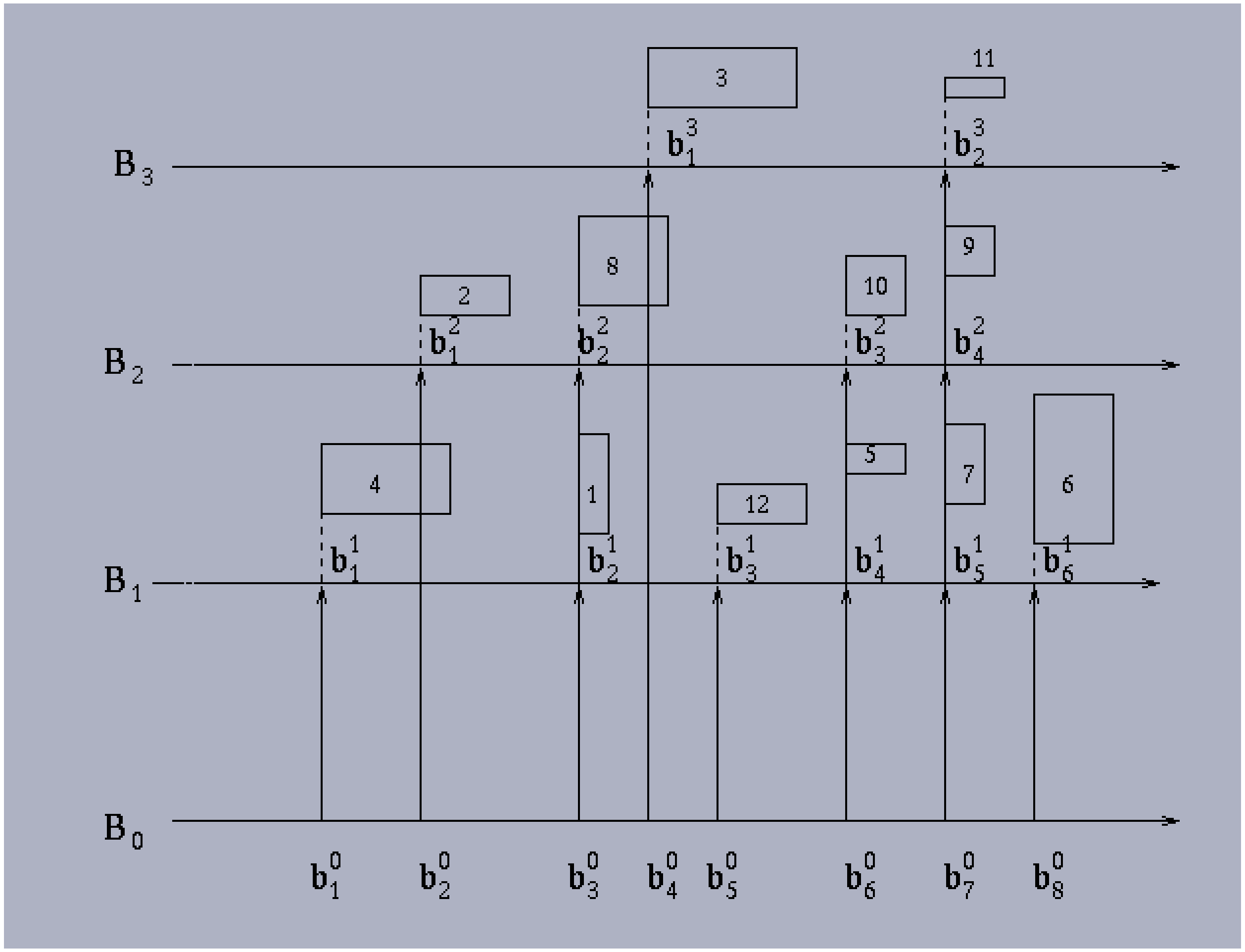


The corresponding graphs of the two dimension classifier

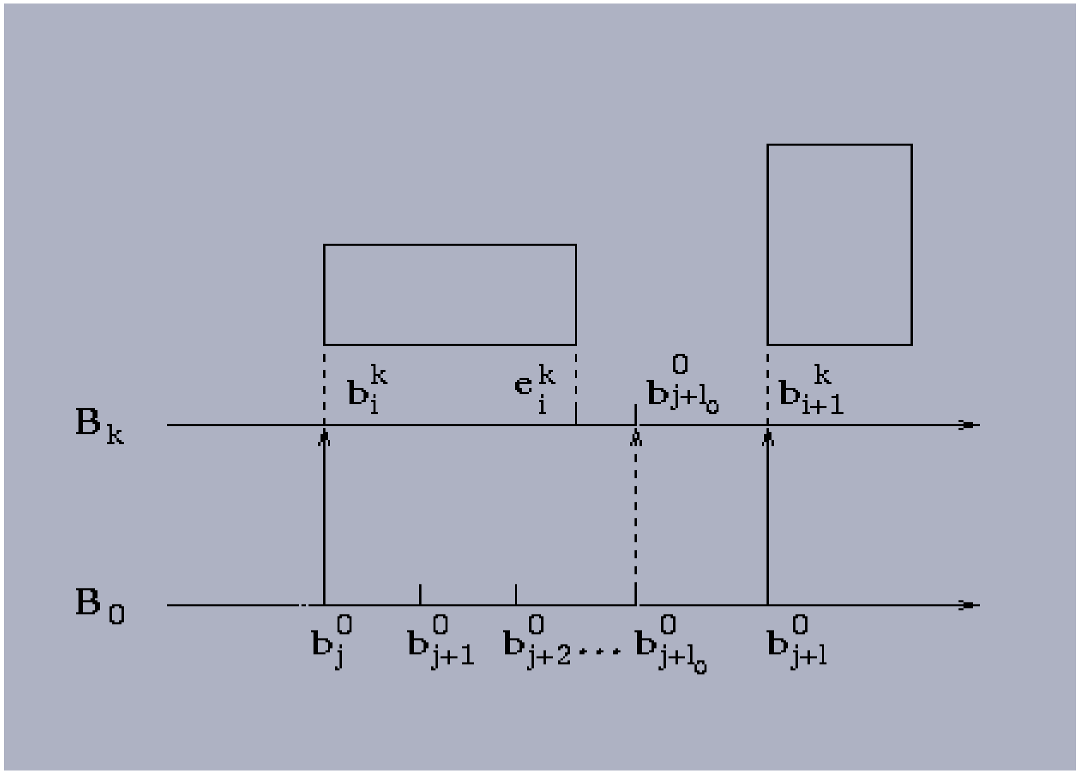




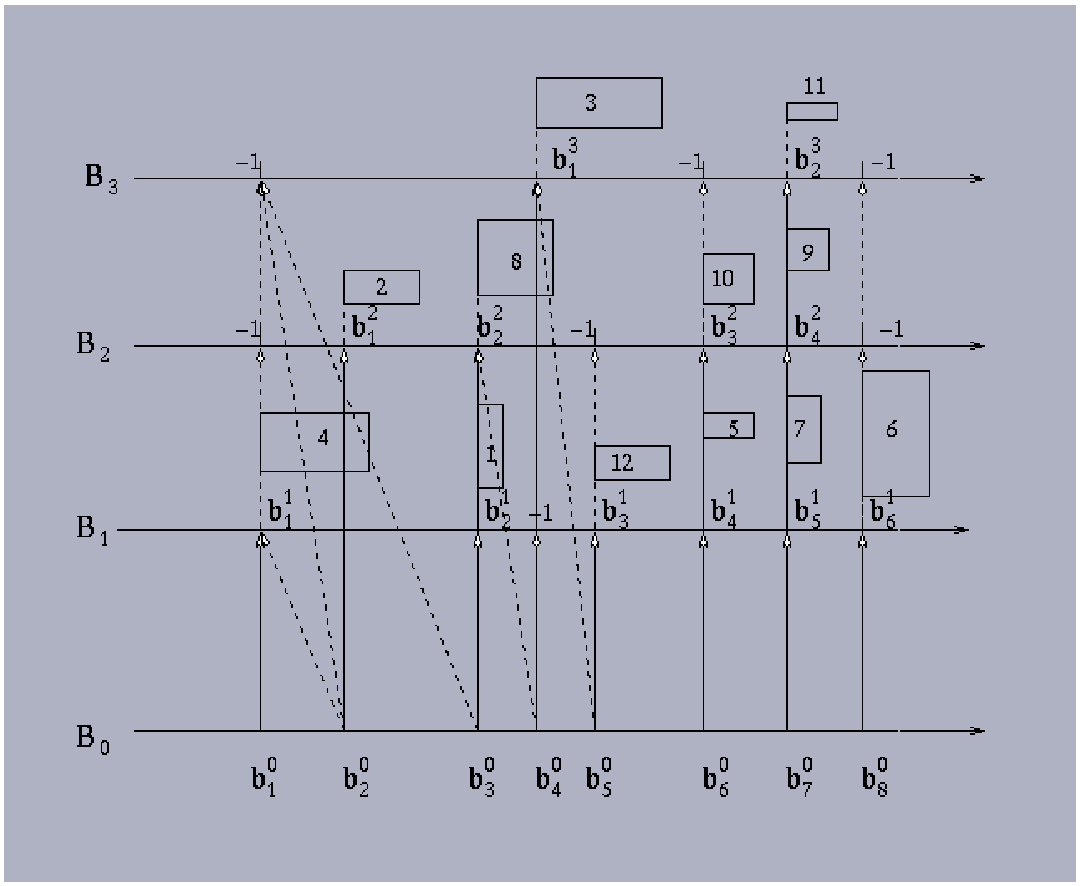
Merge sets of begin points into a master set



Add a virtual point to B_k



A data structure example



$b_1^0 \rightarrow$	4	−1	−1
$b_2^0 \rightarrow$	4	2	−1
$b_3^0 \rightarrow$	1	8	−1
$b_4^0 \rightarrow$	−1	8	3
$b_5^0 \rightarrow$	12	−1	3
$b_6^0 \rightarrow$	5	10	−1
$b_7^0 \rightarrow$	7	9	11
$b_8^0 \rightarrow$	6	−1	−1

Experiments with different repeating factors

f	des	rf	src	rf	I-set
2	15422	1.95	14321	2.09	9
10	2807	10.69	2960	10.14	23
20	1422	21.10	1610	18.63	33
60	489	61.35	498	60.24	74

Experiments with different table sizes

rules	des	rf	rsc	rf	I-set
2000	65	30.8	62	32.3	34
10000	323	31	360	27.8	40
20000	677	29.5	710	28.2	43
100000	3499	28.6	3287	30.4	45
200000	7155	28	6567	30.5	48
1000000	32778	30.5	33698	29.7	61

Experiments with different percent of wildcards

%wildcards	Des rf	Src rf	I-set
10	1.46	1.46	6
30	1.59	1.58	7
50	1.72	1.72	6

Features

- inherently parallel
- very small memory requirements
- neither sensitive to the size of the rule table
- nor the percentage of wildcards in the fields
- scales well from two dimensional classifiers to high dimensional ones
- fast update
- easy to exploit the parallel and pipeline mechanism in the hardware

Thanks