Luleå algorithm

The **Luleå algorithm** of computer science, designed by Degermark et al. (1997), is a technique for storing and searching internet routing tables efficiently. It is named after the Luleå University of Technology, the home institute/university of the technique's authors. The name of the algorithm does not appear in the original paper describing it, but was used in a message from Craig Partridge to the Internet Engineering Task Force describing that paper prior to its publication.[1]

The key task to be performed in internet routing is to match a given IPv4 address (viewed as a sequence of 32 bits) to the longest prefix of the address for which routing information is available. This prefix matching problem may be solved by a trie, but trie structures use a significant amount of space (a node for each bit of each address) and searching them requires traversing a sequence of nodes with length proportional to the number of bits in the address. The Luleå algorithm shortcuts this process by storing only the nodes at three levels of the trie structure, rather than storing the entire trie.

Before building the Luleå trie, the routing table entries need to be preprocessed. Any bigger prefix that overlaps a smaller prefix must be repeatedly split into smaller prefixes, and only the split prefixes which does not overlap the smaller prefix is kept. It is also required that the prefix tree is complete. If there is not routing table entries for the entire address space, it must be completed by adding dummy entries, which only carries the information that no route is present for that range. This enables the simplified lookup in the Luleå trie (Sundström 2007).

The main advantage of the Luleå algorithm for the routing task is that it uses very little memory, averaging 4–5 bytes per entry for large routing tables. This small memory footprint often allows the entire data structure to fit into the routing processor's cache, speeding operations. However, it has the disadvantage that it cannot be modified easily: small changes to the routing table may require most or all of the data structure to be reconstructed. A modern home-computer (PC) has enough hardware/memory to perform the algorithm. The Luleå algorithm is patented in the United States (Degermark et al. 2001).

Contents

First level
Second and third levels
Notes
References

First level

The first level of the data structure consists of

- A bit vector consisting of 2^{16} = 65,536 bits, with one entry for each 16-bit prefix of an IPv4 address. A bit in this table is set to one if there is routing information associated with that prefix or with a longer sequence beginning with that prefix, or if the given prefix is the first one associated with routing information at some higher level of the trie; otherwise it is set to zero.
- An array of 16-bit words for each nonzero bit in the bit vector. Each datum either supplies an index that points to the second-level data structure object for the corresponding prefix, or

supplies the routing information for that prefix directly.
- An array of "base indexes", one for each consecutive subsequence of 64 bits in the bit vector, pointing to the first datum associated with a nonzero bit in that subsequence.
- An array of "code words", one for each consecutive subsequence of 16 bits in the bit vector. Each code word is 16 bits, and consists of a 10-bit "value" and a 6-bit "offset". The sum of the offset and the associated base index gives a pointer to the first datum associated with a nonzero bit in the given 16-bit subsequence. The 10-bit value gives an index into a "maptable" from which the precise position of the appropriate datum can be found.
- A maptable. Because the prefix tree is required to be complete, there can only exist a limited amount of possible 16-bit bitmask values in the bit vector, 678. The maptable rows correspond to these 678 16-bit combinations, and columns the number of set bits in the bitmask at the bit location corresponding to the column, minus 1. So column 6 for the bitmask 1010101010101010 would have the value 2. The maptable is constant for any routing table contents.

To look up the datum for a given address *x* in the first level of the data structure, the Luleå algorithm computes three values:

1. the base index at the position in the base index array indexed by the first 10 bits of *x*
2. the offset at the position in the code word array indexed by the first 12 bits of *x*
3. the value in maptable[*y*][*z*], where *y* is the maptable index from the code word array and *z* is bits 13–16 of *x*

The sum of these three values provides the index to use for *x* in the array of items.

Second and third levels

The second and third levels of the data structure are structured similarly to each other; in each of these levels the Luleå algorithm must perform prefix matching on 8-bit quantities (bits 17–24 and 25–32 of the address, respectively). The data structure is structured in "chunks", each of which allows performing this prefix matching task on some subsequence of the address space; the data items from the first level data structure point to these chunks.

If there are few enough different pieces of routing information associated with a chunk, the chunk just stores the list of these routes, and searches through them by a single step of binary search followed by a sequential search. Otherwise, an indexing technique analogous to that of the first level is applied.

Notes

1. "second Europe trip for IETFers... (http://www1.ietf.org/mail-archive/web/ietf/current/msg01795.html)", Craig Partridge to IETF, May 1, 1997.

References

- Degermark, Mikael; Brodnik, Andrej; Carlsson, Svante; Pink, Stephen (1997), "Small forwarding tables for fast routing lookups", *Proceedings of the ACM SIGCOMM '97 conference on Applications, Technologies, Architectures, and Protocols for Computer Communication* (http://urn.kb.se/resolve?urn=urn:nbn:se:ltu:diva-24248), pp. 3–14, doi:10.1145/263105.263133 (https://doi.org/10.1145%2F263105.263133), S2CID 17232414 (https://api.semanticscholar.org/CorpusID:17232414).

- US 6266706 (https://worldwide.espacenet.com/textdoc?DB=EPODOC&IDX=US6266706), Degermark, Mikael; Andrej Brodnik & Svante Carlsson et al., "Fast routing lookup system using complete prefix tree, bit vector, and pointers in a routing table for determining where to route IP datagrams", issued 2001.
- Medhi, Deepankar; Ramasamy, Karthikeyan (2007), *Network Routing: Algorithms, Protocols, and Architectures*, Elsevier, pp. 510–513, ISBN 978-0-12-088588-6.
- Sundström, Mikael (2007), "Time and Space Efficient Algorithms for Packet Classification and Forwarding", *Doctoral Thesis*, ISSN 1402-1544 (https://www.worldcat.org/issn/1402-1544).
