Randomized Algorithms 2020-1
Lecture 10

The Algorithmic Lovasz Local Lemma and Cuckoo Hashing *

Moni Naor

1 Recap

we reviewed the non-constructive version of the local lemma and its application. We mentioned an approach for making it constructive by Jozsef Beck (see the Alon-Spencer book). We then covered the Moser-Tardos algorithm and its proof [6].

2 Hash Tables

Hash tables is very well studied subject in computer science and one of the more useful practices. Knuth's "The Art of Computer Programming" Volume 3 devotes a lot of space to the various possibilities and the origin of the idea is attributed to a 1956 paper by Arnold Dumey [5]. A major issue is how to resolve collisions and popular suggestions are chaining and Open addressing (or closed hashing). For the latter we need to specify a probing scheme and one of the more popular ones is linear probing which takes advantage of the locality properties of computer memory (in the various levels of hierarchy).

The 'modern' era of investigating hashing can be seen in the work of Carter and Wegman [2] who suggested the idea of thinking of the input as being worst case and the performance is investigated when the hash function is chosen at random from a predefined family (rather than assuming a truly random function).

The simplest way to obtain dictionaries with expected $O(1)$ per operation is to use chained hashing with a table of size $O(n)$. Here, it is enough to choose the hash function from a δ-universal family, for δ which is $O(1/n)$. The *expected length* of a chain is now $O(1)$ and the length of the chain is what determines the cost of an operation. Note however that there will be long chains. Universality on its own only suffices to guarantee that the expected length of the *longest* chain is $O(\sqrt(n))$. The upper bound follows from considering all potential collisions: there are $\binom{n}{2}$ of them. Each collision occurs with probability $O(1/n)$, so the expected number of collisions is $O(n)$. On the other hand,

*These notes summarize the material covered in class, usually skipping proofs, details, examples and so forth, and possibly adding some remarks, or pointers. In the interest of brevity, most references and credits were omitted.

in a chain of length ℓ there are $\binom{\ell}{2}$ collisions. Therefore the expected length of the longest chain cannot be larger than $O(\sqrt{(}n))$. For the lower bound see Alon et al. [1].

What happens if the hash function is truly random? Then this is the "ball and bins" scenario which we will talk in the next lecture and here the heaviest bin/chain is likely to contain $\Theta(\log n/\log\log n)$ elements.

Universal hashing on its own also just guarantees *expected* $O(1)$ performance and not, say, high probability $(1-1/poly(n))$ amortized $O(1)$ performance. There are several ways to obtain this sort of result.

We explored "**Cuckoo Hashing**", suggested by Pagh and Rodler [3], a method that uses two hash functions h_1 and h_2 and where each element x in the set resides either in location $h_1(x)$ or location $h_2(x)$. This means that lookup requires just two accesses to the memory. Insertion may be more involved and requires relocating elements.

Question: when is cuckoo hashing successful? I.e. for a set S and functions h_1 nad $h_@$, when does the insertion algorithm succeed in inserting all the elements?

A lecture be Eric Demaine on hashing is available and recommended [4].

References

[1] Noga Alon, Martin Dietzfelbinger, Peter B. Miltersen, Erez Petrank, and Gabor Tardos, *Linear Hashing* Journal of the ACM, Vol. 46(5), 1999. `http://www.brics.dk/RS/97/16/BRICS-RS-97-16.pdf`

[2] J. L. Carter and M. N. Wegman, Universal classes of hash functions, J. Comput. Syst. Sci. 18 (1979) 143–154. `http://www.cs.princeton.edu/courses/archive/fall09/cos521/Handouts/universalclasses.pdf`

[3] Rasmus Pagh and Flemming Friche Rodler: *Cuckoo hashing*. J. Algorithms 51(2): 122-144 (2004).

[4] Eric Demaine, Lecture 10 in course 6.851: Advanced Data Structures (Spring'12) on Dictionaries, `https://courses.csail.mit.edu/6.851/spring12/lectures/L10.html`

[5] Arnold Isaac Dumey, *Indexing for rapid random-access memory*, Computers and Automation 5 (12), 6–9, 1956.

[6] R. Moser and G. Tardos, *A constructive proof of the general lovász local lemma*, J. ACM, February 2010 Article No.: 11.