TIGHT BOUNDS ON THE COMPLEXITY OF THE BOYER-MOORE PATTERN MATCHING ALGORITHM. pdf

1: Information Retrieval: CHAPTER STRING SEARCHING ALGORITHMS

() Tight bounds on the complexity of the Apostolico-Giancarlo algorithm. Information Processing Letters , Richard Cole and Ramesh Hariharan.

The Karp-Rabin algorithm In practice, this algorithm is slow due to the multiplications and the modulus operations. However, it becomes competitive for long patterns. We can avoid the computation of the modulus function at every step by using implicit modular arithmetic given by the hardware. In other words, we use the maximum value of an integer determined by the word size for q. The value of d is selected such that dk mod 2r has maximal cycle length cycle of length 2r-2 , for r from 8 to 64, where r is the size, in bits, of a word. For example, an adequate value for d is With these changes, the evaluation of the signature at every step see Figure In this way, we use two multiplications instead of one multiplication and two modulus operations. The results are similar. For the shift-or algorithm, the given results are for the efficient version. The results for the Karp-Rabin algorithm are not included because in all cases the time exceeds seconds. For this reason, if the pattern is small 1 to 3 characters long it is better to use the naive algorithm. If the alphabet size is large, then the Knuth-Morris-Pratt algorithm is a good choice. In all the other cases, in particular for long texts, the Boyer-Moore algorithm is better. Finally, the Horspool version of the Boyer-Moore algorithm is the best algorithm, according to execution time, for almost all pattern lengths. The shift-or algorithm has a running time similar to the KMP algorithm. However, they are not space optimal in the worst case because they use space that depends on the size of the pattern, the size of the alphabet, or both. Galil and Seiferas , show that it is possible to have linear time worst case algorithms using constant space. See also Slisenko [,]. They also show that the delay between reading two characters of the text is bounded by a constant, which is interesting for any real time searching algorithms Galil Practical algorithms that achieve optimal worst case time and space are presented by Crochemore and Perrin , Optimal parallel algorithms for string matching are presented by Galil and by Vishkin See also Berkman et al. Simulation results for all the algorithms in English text Many of the algorithms presented may be implemented with hardware Haskin ; Hollaar For example, Aho and Corasick machines see Aoe et al. If we allow preprocessing of the text, we can search a string in worst case time proportional to its length. This is achieved by using a Patricia tree Morrison as an index. This solution needs O n extra space and O n preprocessing time, where n is the size of the text. For other kinds of indices for text, see Faloutsos , Gonnet , Blumer et al. For further references and problems, see Gonnet and Baeza-Yates Perspectives and Open Problems, ed. Also as Research Report CS To appear in Communications of ACM. Lecture Notes in Computer Science , pp. Theorefical Computer Science, 40, Lecture Notes in Computer Science Paris 7 submitted for publication. Information Processing Letters, pp. To appear in Software-Practice and Experience. The C Programming Language.

TIGHT BOUNDS ON THE COMPLEXITY OF THE BOYER-MOORE PATTERN MATCHING ALGORITHM. pdf

2: Boyer-Moore algorithm

The Aho-Corasick algorithm uses a finite state machine for string matching and provides the best worst case performance, but requires a significant amount of memory for the state machine.

Richard Cole, Shravas Rao. Richard Cole, Yixin Tao. Richard Cole, Jose R. Correa, Vasilis Gkatzelis, Vahab S. Decentralized utilitarian mechanisms for scheduling games. Games and Economic Behavior Inner product spaces for MinSum coordination mechanisms. Amortized Analysis on Asynchronous Gradient Descent. Richard Cole, Tim Roughgarden. The sample complexity of revenue maximization. Full paper with improved results arxiv: Tatonnement Beyond Gross Substitutes? Gradient Descent to the Rescue. Mechanism Design for Fair Division: Allocating Divisible Items without Payments. Positive results for mechanism design without money. Tatonnement in ongoing markets of complementary goods. Prompt Mechanisms for Online Auctions. Fast-converging tatonnement algorithms for one-time and ongoing market problems. Bottleneck links, variable demand, and the tragedy of the commons. How much can taxes help selfish routing? Pricing network edges for heterogeneous selfish users. Resource Oblivious Sorting on Multicores. Bender, Richard Cole, Erik D. Maintaining Data for Traversals in a Memory Hierarchy. Bender, Richard Cole, Rajeev Raman. Suffix Trays and Suffix Trists: Structures for Faster Text Indexing. ACM Transactions on Algorithms 11 2: Parallel Two Dimensional Witness Computation. Preliminary version appeared as part of: Optimally fast parallel algorithms for preprocessing and pattern matching in one and two dimensions. Algorithms, Applications, and a Lower Bound. Preliminary version appeared in STOC Verifying candidate matches in sparse and wildcard matching. A Simpler Faster Algorithm. Preliminary version appeared in SODA Faster approximate string matching. Which patterns are hard to find. Israeli Symposium on Theoretical Computer Science, Tight bounds on the complexity of the Boyer--Moore string matching algorithm. Tighter upper bounds on the exact complexity of string matching. See also FOCS, ,

TIGHT BOUNDS ON THE COMPLEXITY OF THE BOYER-MOORE PATTERN MATCHING ALGORITHM. pdf

3: Complexity of sequential pattern matching algorithms | Wojciech Szpankowski - www.enganchecubano.

Tight bounds on the complexity of the Boyer-Moore string matching algorithm. pattern matching by Boyer-Moore-type algorithms, Proceedings of the sixth annual ACM.

Computer Science, Purdue University, W. We formally de ne a class of sequential pattern matching algorithms that includes all variations of Morris-Pratt algorithm. For the last twenty years it was known that the complexity of such algorithms is bounded by a linear function of the text length. Recently, substantial progress has been made in identifying lower bounds. We now prove there exists asymptotically a linearity constant for the worst and the average cases. We use Subadditive Ergodic Theorem and prove an almost sure convergence. Our results hold for any given pattern and text and for sta- tionary ergodic pattern and text. This property seems to be uniquely reserved for Morris-Pratt type algorithms e. Nevertheless, the existence of such a constant has never been proved. The only exception known to us is the average case of Morris-Pratt-like algorithms [18] cf. In this paper we investigate a fairly general class of algorithms, called sequen- tial algorithms, for which the existence of the linearity constant in an asymptotic sense is proved for the worst and the average case. Sequential algorithms in- clude the naive one and several variants of Morris-Pratt algorithm [16]. These algorithms never go backward, work by comparisons, and are easy to implement. They perform better than Boyer-Moore like algorithms in numerous cases, e. Thus, even from a prac- tical point of view these algorithms are worth studying. In this paper we analyze sequential algorithms under a general probabilistic model that only assumes stationarity and ergodicity of the text and pattern sequences. We show that asymptotic complexity grows linearly with the text length for all but nitely many strings i. The proof relies on the Subadditive Ergodic Theorem [11]. The literature on worst case as well average case on Knuth-Morris-Pratt type algorithms is rather scanty. For almost twenty years the upper bound was known [16], and no progress has been reported on a lower bound or a tight bound. This was partially recti ed by Colussi et al. In the course of proving our main result, we construct the so called unavoidable positions where the algorithm must stop to compare. The existence of these positions is crucial for establishing the subadditivity of the complexity function for the Morris-Pratt type algorithms, and hence their linearity. This property seems to be restricted to Morris-Pratt type algorithms e. The paper is organized as follows. In the next section we present a general de nition of sequential algorithms, and formulate our main results. Section 3 contains all proofs. Then, we formulate our main results and discuss some consequences. We also assume that for a given pattern p its length m does not vary with the text length n. Our prime goal is to investigate complexity of string matching algorithms that work by comparisons i. We assume in the following that this comparison is performed at most once. If either the pattern or the text is a realization of a random sequence, then we denote the complexity by a capital letter, that is, we write Cn instead of cn. Our goal is to nd an asymptotic expression for cn and Cn for large n under deterministic and stochastic assumptions regarding the strings p and t. For simplicity of notation we often write cn instead of cn t; p. We need some further de nitions that will lead to a formal description of sequential algorithms. We start with a de nition of an alignment position. Intuitively, at some step of the algorithm, an alignment of pattern p at po- sition AP is considered, and a comparison is made with character p[k] of the pattern. Finally, we are ready to de ne sequential algorithms. Sequentiality refers to a special structure of a sequence of positions that pattern and text visit during a string matching algorithm. A string searching algorithm is said to be: Note that our assumptions on non-decreasing text positions in ii implies i. Furthermore, non-decreasing alignment positions implies that all oc- currences of the pattern before this alignment position were detected before this choice. Although 2 is not a log- ical consequence of semi-sequentiality, it represents a natural way of using the available information for semi-sequential algorithms. In that case, subpattern tll 1k 1 is known when t[l] is read. There is no need to compare p[k] with t[l] if tll 1k 1 is not a pre x of p of size k 1, i. We now illustrate our de nition on several examples. Naive or brute force algorithm The simplest string searching algorithm is the naive one. All text

positions are alignment positions. For a given one, say AP , text is scanned until the pattern is found or a mismatch occurs. This algorithm is sequential hence semi-sequential but not strongly sequen- tial. Morris-Pratt-like algorithms [16, 19]. Di erent choices lead to di erent variants of the classic Morris-Pratt algorithm [16]. They di er by the use of the information obtained from the mismatching position. We formally de ne three main variants, and provide an example. One de nes a shift function S to be used after any mismatch as: Illustration to De nition 3. The rst mismatch occurs for M 12; The comparisons performed from that point are: Some observations are in sequel: Morris-Pratt variant considers one align- ment position at a time, while the optimal sequential algorithm, that of Simon, considers several alignment positions at the same time, and may disregard sev- eral of them simultaneously e. It is interesting to observe that the subset f1; 5; 12g of alignments positions appears in all variants. Our de nition of semi-sequentiality is very close to the de nition of sequen- tiality given in [13]. The on-line algorithms are very close to our strongly sequential ones. Finally, in the course of proving our main result we discover an interesting structural property of sequential algorithms that we already observed in Ex. Having in mind the above de nitions we can describe our last class of se- quential algorithms containing all variants of KMP-like algorithms for which we formulate our main results. In passing we note that the naive pattern matching algorithm cf. We prove below that all strongly sequential algorithms i. Before, however, we must describe modeling assumptions concerning the strings. We adopt one of the following assumptions: B Semi-Random model The text string t is a realization of a stationary and ergodic sequence while the pattern string p is given. C Stationary Model Strings t and p are realizations of a stationary and ergodic sequence cf. Formulation of our results depends on the model we work with. In the de- terministic model we interpret the complexity cn t; p as the worst case com- plexity i. Under assumption B we consider almost sure a. Finally, in the stationary model C we use standard average case complexity, that is, ECn. Now we are ready to formulate our main results. Let p be a given pattern of length m. If Et denotes the the average cost over all text strings, the following also holds: Finally, with respect to our main class of algorithms, namely, Morris-Pratt like i. Strongly sequential algorithms e. But, in this paper we prove that such a uctuation cannot take place for the complexity function of the strongly sequential pattern matching algorithms. The idea of the proof is quite simple. We shall show that a function of the complexity i. In passing, we point out that the most challenging is establishing the subadditivity property to which most of this section is devoted. Thus, to prove our main results we need to establish the subadditivity prop- erty for the complexity cn t; p for all texts t and patterns p. Let Ur be the smallest unavoidable position greater than r. This part involves the following contributions: We call this contribution S1. Observe that those compar- isons contribute to c1;n but not to c1;r. To avoid counting the last character r twice, we must subtract one comparison. The following contributions must be considered: Illustration to the proof of Lemma 11 To complete the proof, it remains to nd upper bounds on S1 , S2 , and S3. So are their di erence. It relies on Theorem 8 so we present the proof of Theorem 8 rst. Let fAi g be the set of alignment positions de ned by a strongly sequential algorithm that runs on tnr. As it contains r, we may de ne cf. Let the adversary assumes that p does occur. This completes the proof of Theorem 8. Illustration to the proof of Theorem 8 Finally, we turn to the proof of Theorem 9.

TIGHT BOUNDS ON THE COMPLEXITY OF THE BOYER-MOORE PATTERN MATCHING ALGORITHM. pdf

4: pattern matching algorithms | Download eBook pdf, epub, tuebl, mobi

The problem of finding all occurrences of a pattern of length m in a text of length n is considered. It is shown that the Boyer-Moore string matching algorithm performs roughly 3n comparisons and.

We show how to speed up two string-matching algorithms: The RF algorithm is based on factor graphs for the reverse of the pattern. The main feature of both algorithms is that they scan The main feature of both algorithms is that they scan the text right-to-left from the supposed right position of the pattern. The BM algorithm goes as far as the scanned segment factor is a suffix of the pattern. The RF algorithm scans while the segment is a factor of the pattern. Both algorithms make a shift of the pattern, forget the history, and start again. The RF algorithm usually makes bigger shifts than BM, but is quadratic in the worst case. We show that it is enough to remember the last matched segment represented by two pointers to the text to speed up the RF algorithm considerably to make a linear number of inspections of text symbols, with small coefficient , and to speed up the BM algorithm to make at most 2. Only a constant additional memory is needed for the search phase. We give alternative versions of an accelerated RF algorithm: The paper demonstrates the techniques to transform algorithms, and also shows interesting new applications of data structures representing all subwords of the pattern in compact form. We then use this taxonomy to examine the options for the various components of a search algorithm, and compare We then use this taxonomy to examine the options for the various components of a search algorithm, and compare their execution speeds on a variety of architectures for a natural language test set. As a result, we develop new algorithms that execute more rapidly than those previously known. Theoretical work on string searching has focused on the worst case computational complexity and the asymptotic search behavior for long strings by counting the number of character comparisons made. For the problem of searching for all occurrences of a string of m characters in a text of n characters, we denote the number of text comparisons made by an algorithm in the worst case as c n ,m. This paper shows how to modify their algorithm to use fewer comparisons. Given any fixed ffl? The pattern preprocessing step also takes linear time and uses constant space. We study the exact number of symbol comparisons that are required to solve the string matching problem and present a family of efficient algorithms. Unlike previous string matching algorithms, the algorithms in this family do not "forget" results of comparisons, what makes their analysi The pattern preprocessing takes linear time and makes at most 2m comparisons. This algorithm establishes that, in general, searching for a long pattern is easier than searching for a short one. Efficient String Algorithmics by Dany Breslauer , " Problems involving strings arise in many areas of computer science and have numerous practical applications. We consider several problems from a theoretical perspective and provide efficient algorithms and lower bounds for these problems in sequential and parallel models of computation. In the sequential setting, we present new algorithms for the string matching problem improving the previous bounds on the number of comparisons performed by such algorithms. In parallel computation, we present tight algorithms and lower bounds for the string matching problem, for finding the periods of a string, for detecting squares and for finding initial palindromes.

TIGHT BOUNDS ON THE COMPLEXITY OF THE BOYER-MOORE PATTERN MATCHING ALGORITHM. pdf

5: Apostolicoâ€“Giancarlo algorithm - Wikipedia

String matching is the problem of finding all the occurrences of a pattern in a text. We present an new method to compute the combinatorial shift function ("matching shift") of the well-known Boyer-Moore string matching algorithm.

In this paper we study the exact comparison complexity of the string prefix-matching problem in the deterministic sequential comparison model with equality tests. We derive almost tight lower and upper bounds on the number of symbol comparisons required in the worst case by on-line prefix-matchi We derive almost tight lower and upper bounds on the number of symbol comparisons required in the worst case by on-line prefix-matching algorithms for any fixed pattern and variable text. Unlike previous results on the comparison complexity of string-matching and prefix-matching algorithms, our bounds are almost tight for any particular pattern. We also consider the special case where the pattern and the text are the same string. This problem, which we call the string self-prefix problem, is similar to the pattern preprocessing step of the Knuth-Morris-Pratt stringmatching algorithm that is used in several comparison efficient stringmatching and prefix-matching algorithms, including in our new algorithm. We obtain roughly tight lower and upper bounds on the number of symbol comparisons required in the worst case String matching is the problem of finding all the occurrences of a pattern in a text. We present an new method to compute the combinatorial shift function "matching shift" of the well-known Boyer-Moore string matching algorithm. This method implies the computation of the length of the lo This method implies the computation of the length of the longest suffixes of the pattern ending at each position in this pattern. These values constituted an extra-preprocessing for a variant of the Boyer-Moore designed by Apostolico and Giancarlo. We give here a new presentation of this algorithm together with a tight bound of 1. Military Communications Conference , " Unfortunately, this delay becomes more significant as security policies Unfortunately, this delay becomes more significant as security policies and network speeds continue to increase. This paper introduces a new parallel IDS content matching technique that provides initial packet inspections with less delay. The technique distributes portions of a packet payload across an array of n processors, each responsible for scanning a smaller amount of original payload. Given this design, each processor has less data to inspect thus reducing the overall delay. Unlike similar parallel approaches, our technique ensures that security is maintained no false negatives. Furthermore, the proposed parallel technique is shown to result in an initial match speed-up of approximately 1. Show Context Citation Context The Wu-Manber algorithm can be viewed as a multi-pattern version of Boyer-Moore, which requires less memory than Aho-Corasick and provides a better average case performance. The Apostolico-Giancarlo string-matching algorithm is analyzed precisely. We give a tight upper bound of 3 2 n text characters comparisons when searching for a pattern in a text of length n. We exhibit a family of patterns and texts reaching this bound. We also provide a slightly improved version of We also provide a slightly improved version of the algorithm. This paper shows that Generalized Partial Computation GPC can automatically generate efficient string matching algorithms. GPC is a program transformation method utilizing partial information about input data and auxiliary functions as well as the logical structure of a source program. GPC uses both a classical partial evaluator and an inference engine such as a theorem prover to optimize programs. First, we show that a Boyer-Moore BM type pattern matcher without the bad-character heuristic can be generated from a simple non-linear backward matcher by GPC. This sort of problems has already been discussed in the literature using offline partial evaluators. However, there was no proof that every generated matcher runs in the same way as the BM. In this paper we prove that the problem can be solved starting from a simple non-linear pattern matcher as a source program. Theory and Practice by Ricardo A. Baeza-yates, Gonzalo Navarro " We present the state of the art of the main component of text retrieval systems: We outline the main lines of research and issues involved. We survey the relevant techniques in use today for text searching and explore the gap between theoretical and practical algorithms.

TIGHT BOUNDS ON THE COMPLEXITY OF THE BOYER-MOORE PATTERN MATCHING ALGORITHM. pdf

The main observation is that simpler ideas are better in practice. This was thesrst algorithm that did not compare all the text characters. In practice, the algorithm was rather complicated to implement, not to mention if one has to ensure linear worst-case complexity. A practical improvement was obtained by noting that, for large enoug We study the complexity of a classical combinatorial problem of computing the period of a string. We investigate both the average- and the worst-case complexity of the problem. We deliver almost tight bounds for the average-case.

6: Richard Cole home page

The Boyer-Moore's string matching algorithm uses two pre-computed tables skip, which utilizes the occurrence heuristic of symbols in a pattern, and shift, which utilizes the match heuristic of the pattern, for searching a string.

7: Boyerâ€“Moore string-search algorithm - Wikipedia

Tight bounds on the complexity of the Boyer-Moore pattern matching algorithm Item Preview remove-circle Share or Embed This Item.

TIGHT BOUNDS ON THE COMPLEXITY OF THE BOYER-MOORE PATTERN MATCHING ALGORITHM. pdf

Monitoring hospital patients using ambient displays Monica Tentori, Daniela Segura, and Jesus Favela Definition of strategic human resource management The new philosophy : giving up the crystalline purity of logic 2017 range rover sport hse owners manual Pnb home loan application form Advances in Ceramic Matrix Composites XI (Ceramic Transactions Series) Karma of materialism Gender dominance short stories Progress or Whitneyism, which? Bells picturesque guide to American watering places. Traditional Values In Action Family Home Extensions of the standard template V. 3. Vietnam, Singapore, and Central Thailand. The Evolutionary Bases of Consumption (Marketing and Consumer Psychology Series) Hard act to follow Chinas new diplomatic offensive. A Guide to current policies and practices. Rc car magazine Yearbook of legslation, 1903- The world economy in transition Discrete probability models and methods Oriental Erotic Art Sewing patterns for baby boy Handy Home Medical Advisor Con M Augustine, Pascal, and Hume for the postmodern world? A phenomenology of judgment: Charles Reznikoffs Holocaust Lectures on systematic theology and pulpit eloquence. Dont tell the girls Religious Emblems The Athena treasury The Gayman/Gehman/Gahman family Aboard the Ship Great Republic to New Orleans Crochet scrunchie Gas situation in the ECE region around the year 1990 Theres a Cow in My Swimming Pool Witches night out Places of passage, places of fear A Review of Options Ccna 640 802 ebook 17 Seven Brides for Seven Brothers 189