1: OVERVIEW

1.1 Mnemonic codes for numbers
Symbol sequences of arbitrary length and content are hard for us to remember, so we need a system that aids in the task of memorization. Mnemonics are devices or techniques that are used to aid in memorizing something. They help in associating the material to be remembered to different sense modes, and hence improve its recall.

In this research we report on the generalization of a particular mnemonic, known as katapayadi in medieval Indian literature [Sankaravarman1823, Raman97] and defined in Section 2.2, for applications in diverse areas of computing. In particular, we present novel schemes to (i) organize information for its effective retrieval, (ii) provide a cover for the information to hide it from intruders' eyes, (iii) facilitate authentic users in generating and managing mnemonic secure passwords and (iv) generate mnemonic text for telephone numbers.

The original katapayadi scheme was intended for humans. Exploring the potential—the power, versatility and scope—of the generalized katapayadi scheme when man and machine interact co-operatively forms an important aspect of the subject matter of this work. We call this scenario an “interactive man-machine environment” (IMME).

In its original form, the katapayadi system aids us in remembering numbers. It is an empirical fact that remembering long sequences of digits is not easy for most of us. Numbers are an unavoidable part of life, more so in the modern age. Very often we come across small and big numbers as a result of counting and arithmetic, and nowadays due to various numeric identification codes. Research in psychology has established that normal human beings can manage to remember arbitrary sequences of 7 ± 2 digits, while longer sequences rapidly become harder to retain [Miller56]. Take π, the ratio of the circumference of a circle to its diameter, which has the value 3.141582365 ... . This infinitely long sequence of digits is one of the most fundamental constants of nature; it has no recurring or other interesting pattern that would enable us to remember its most significant digits. In the next Chapter, we demonstrate how values like π can be remembered by humans, to arbitrarily long precision.
1.2 The katapayadi mnemonic code

The katapayadi mnemonic code is a scheme that enables us to embed large numbers into interesting word sequences which can be easily remembered. The original katapayadi association map is a one-to-many mapping from digits to Sanskrit consonants. With such a map, a digit can be encoded by any one of the 3 or 4 consonants associated with it, and hence—assuming that vowels are superfluous—a sequence of digits can be encoded by one or more words which convey meaning, i.e., contain information. Research in psychology confirms what we know intuitively: the human brain performs faster registration and effective retrieval of information if it is interesting and involves concrete objects rather than abstractions like numbers [Bower70, Yan00]. This is the basis of the efficacy of the katapayadi encoding.

Details of the original katapayadi scheme and a generalization of it are given in Chapter 2, and further variations are found in later Chapters. The generalization has two aspects. First, the katapayadi mapping is from any arbitrary alphabet—natural or formal—to any other alphabet, the range is arbitrarily partitioned into so-called "consonants" and "vowels" and then the procedures of the original scheme are employed. Second, there are user-specified heuristic rules for the construction and application of the katapayadi association maps. This generalization has been named k-logic for the sake of brevity and simplicity. Accompanied by other application-specific rules or modifications, k-logic has been used extensively throughout the thesis. (In general, the prefix "k-" indicates the extension or modification of an idea in the context of generalized katapayadi, e.g., k-association map or k-steganography.)

As stated above, the applications of katapayadi to several areas of computing form the subject matter of this research. All these applications share a common, higher-order attribute, namely, they share an interactive man-machine interactive environment. The IMME models presented in this work are intended to utilize the intelligence of the participating human user; the machine is to facilitate him, e.g., by providing easy access to the required data. Human and machine interact cooperatively and in a short time converge to a solution that satisfies the user. The novelty of the present approach is in its deviation from conventional AI techniques, where the machine takes over the functions of the human. One consequence of IMME is that the required programming is greatly simplified.

1. Overview
In each application domain, generalization of the basic katapayadi scheme takes into account the specific requirements of the application. In every instance, the fundamental feature of the katapayadi encoding—its mnemonic property—is put to good use. Additional features which provide katapayadi the power of both flexibility and complexity are: its homophonic property, programmability or algorithmic potential, and map-key based organization of the association map. These ideas are discussed in detail in subsequent Chapters.

1.3 Thesis organization
The thesis is organized as follows: Chapter 2 describes the concepts underlying katapayadi, while the subsequent Chapters spell out how it is used in diverse application domains. Chapters 3 to 6 are organized in pairs: first the current state of research in a domain is summarized, and in the next the contributions of the present research are spelt out. Chapters 7 and 8 form a pair, though in a different way. Chapter 7 contains a detailed analysis of the data structures and retrieval mechanisms associated with k-logic, and some of the observations of this analysis are employed in Chapter 8. The important results of our research are compiled in Chapter 9 and some directions for the future work are discussed. Finally, the history and evolution of the katapayadi scheme is given in an Appendix.

The contents of the thesis are summarized Chapterwise in Sections 1.4 to 1.12 below. (Note: Citations to the original research papers are given in the respective Chapters; in this Overview only a few papers are cited.)

1.4 The katapayadi mnemonic encoding scheme and its interpretations (Chapter 2)
In Chapter 2 the original katapayadi encoding is defined formally, and several instances of it are given to show how it has served human needs over the centuries. Among these examples is the encoding of $\pi$, mentioned above. In order to highlight the particular advantages inherent in the katapayadi scheme, comparable mnemonic encodings prevalent in other cultures are also described. In the generalization named k-logic, any arbitrary (natural or formal) alphabet can be used for the domain or the range (in place of the original digits and Sanskrit, respectively). Several instances of k-logic are given, including an encoding of $\pi$ in the English language.

Three interpretations of generalized katapayadi are given, for subsequent use in different application domains. The first two have a bearing on security, and the third is
related to information storage and retrieval. First, k-logic is treated as a homophonic substitution cipher. This provides a cryptographic base for the algorithmic construction of steganograms. Second, k-logic as a language for secure communication addresses higher level issues like user-friendliness, programmability and safety. Finally, k-logic is treated as “hashing suited to human processors”. K-logic is compared and contrasted with conventional ("machine-friendly") hashing as well as the method of loci, an established human-friendly information retrieval method.

1.5 Steganography: background and literature survey (Chapter 3)

Steganography—the discipline of hiding information in other information—has been an active area for centuries. Linguistic steganography is concerned with hiding a message in a text cover. As a discipline of computer science it is a relatively new area, with the literature dating from the 1980’s and regular conferences being held from about 1995. However, early examples of the application of these techniques, e.g., watermarking, predate these milestones by a few decades.

In Chapter 3 we start with some basic definitions needed in steganography. Since our research is mostly concerned with linguistic steganography, much of the terminology is concerned with this specialty. (However, for the sake of completeness and to understand the very broad scope of steganography in general, we have also enumerated diverse practical instances from pre-computer days.) Important theoretical and technical issues of steganography, including quality metrics like stealth, robustness, data rate and entropy, are briefly explained. The Chapter ends with a brief enumeration of steganographic systems reported in the literature.

1.6 Application of generalized katapayadi to steganography (Chapter 4)

In conventional steganography a message is hidden in a cover by a sequence of choices: each choice being the selection of a particular word out of a set of synonyms (abbreviated to “synset”) in the cover language. In k-steganography, the meaning of “synonym” is different—synonyms are no longer words with the same meaning in the cover language, but ones having the same image under the k-association map. (We have not coined new terms for the different meanings of “synonym” and “synset”. The semantics of these terms is clearly different here, but their function—interchangeability—remains the same.) Constructing steganograms by choosing
between words having widely varying cover-language meanings needs human intervention; IMME reduces the human burden while retaining stealth.

With this starting point, two schemes are proposed for generating \(k\)-steganograms. They are named BK and GK, since they respectively use the basic and generalized versions of the \(k\)-association map. In BK the text cover is generated for numeric data (hence a non-numeric message is enciphered as a numeric string during a preprocessing phase), whereas GK does not have this constraint. Several examples of \(k\)-steganograms are constructed to illustrate, among other things, the need for and utility of user-specified heuristics. For constructing \(k\)-association maps, it is necessary for the cardinalities of the plaintext symbol set and ciphertext consonant set to have a certain minimum ratio, but this is not always possible in a given situation. The user-specified heuristics get around this difficulty.

A quality analysis of \(k\)-steganograms is given. Their code-breaking strength is computed by a combinatorial analysis of \(k\)-association maps and shown to be very high, given the naïve assumption that a brute-force code-breaking method would have to generate all possible maps on the given alphabet. It is demonstrated that without knowing the key it is almost as hard to reconstruct key-based \(k\)-association maps as it is with randomly generated ones, even if their past usage history is known. It is also shown that rule-based systems using \(k\)-logic perform better than the well-known Baconian cipher (which uses a similar kind of message-embedding). Finally, quality metrics for steganography, discussed in the previous Chapter, are considered for the special case of \(k\)-steganography, and it is argued this approach generates rugged, hard-to-break steganograms.

1.7 Secure password generation systems: background and literature survey (Chapter 5)

The need for secure and user-friendly passwords is explained, and their characteristic features are described. To be strong, i.e., resistant to cracking, passwords need to be long and cryptic, which makes them harder to remember. Random passwords display these advantages as well as disadvantages. People need to remember a large number of passwords, and to ensure security these have to be changed periodically, which most users are reluctant to do. Hence there is a need for user-friendly password generators that address and resolve these issues.
The nature and types of password vulnerabilities are discussed. Password encryption of various kinds can be employed, but for many of these, decrypters are available on the Internet. Different methods of password-cracking attacks—both traditional and newer ones—are briefly mentioned. Robust decryption algorithms are resistant to brute-force attacks, but their efficacy is often compromised because decision attacks exploit weaknesses in the user's choice of passwords. The results of surveys describing users' choice of passwords are summarized. A few techniques for selecting good passwords are reviewed, and their merits and demerits discussed. Password evaluator software (which estimates their strength) is also mentioned.

The entropy of passwords, i.e., the quantitative measure of their strength, is discussed. The entropy per character of different types of passwords and the cost of breaking them are summarized. The entropy of passwords can be increased by "salting", i.e., adding machine generated strings to user-supplied passwords.

Standards for automatic password generators specify, among other things, that passwords should be pronounceable. Attempts have been made to create such password generators using various heuristics but despite claims to the contrary, users find that the resulting passwords are not always easily pronounceable. Mnemonic password generators based on regular expressions is yet another approach, but if the formula for the regular expression is compromised, then the whole system is at risk. Based on these considerations, commercial as well as free password generators are available. The literature survey reports on the features, advantages and disadvantages of automatic password generators. Support for multiple passwords is limited to password database management systems, but they are not coupled to password generators. Proactive password checkers are a different approach—they check if a user-generated password is secure.

Before concluding the Chapter, there is a brief discussion of advanced password technologies, including smart cards, biometric signatures and picture passwords.

1.8 Application of generalized katapayadi to password management (Chapter 6)
The main research contributions are: first, a pronounceable password generator based on a modified definition of "syllable" (the unit of pronunciation), second, an integrated password management scheme based on k-logic that invokes the pronounceable password generator, and third, user-salted passwords (as contrasted to
conventional system-salted passwords). The pronounceable password generator is named the P-scheme and the integrated password management scheme based on k-logic is named the K-scheme.

The P-scheme changes the definition of "syllable" in order to be consistent with the conventions of the katapayadi encoding. Instead of the normal linguistic definition of a pronounceable string as a sequence of syllables where a syllable is defined as onset + nucleus + coda, the new definition treats a pronounceable string as an alternating sequence of valid vowel clusters and consonant clusters. (A consonant/vowel cluster is "valid" if it occurs as part of at least one word in a dictionary.) At the outset, a database of valid initial, medial and final clusters of vowels as well as consonants is constructed after scanning a standard dictionary. The P-scheme constructs pronounceable strings by concatenating randomly selected alternating vowel and consonant clusters, which are supplied to the user to be treated as passwords. The entropy per cluster is computed for initial, medial and final vowel/consonant clusters, and using these figures, the P-scheme generates pronounceable passwords of a given entropy and length.

The K-scheme generates a family of homophonically equivalent passwords using a mapkey and a "passkey" as its parameters. The mapkey that generates the k-association map serves as the user's signature and thus generates a personalized family of passwords, to be used whenever either multiple passwords or periodical password updates are needed. The mapkey also serves the purpose of user authentication when a password reset is requested. The passkey places restrictions on letter combinations that can occur in the password. (This approach is resilient to social engineering attacks.) A method for generating a password family of the required entropy is given, along with examples. An analysis shows that the entropy per character of this scheme is better than that of Diceware, a popular pronounceable password generator.

"Salt" is the addition of a string to the user-supplied password, to make it more secure. In our scheme, the salt is added by the user, in contrast to the conventional system-specified salt. Moreover, the salt is different every time the password is supplied, thus making it even more resilient. This approach increases the entropy of the passwords and resists social-engineering attacks. A novel advantage of user-
supplied salt is that it dynamically facilitates resource access restrictions at the time of password authorization. An algorithm for generating user-salted passwords is given and its commercial applications are discussed.

1.9 K-probing: a retrieval-efficient methodology (Chapter 7)
In Chapter 2, several distinct interpretations of k-logic were presented. One of them was that k-logic may be viewed as "hashing suited to human processors". To make it distinct from the machine variety while capturing some of its essential features, we coined the term "k-probing". In subsequent Chapters, this notion was implied or implicated in the application of k-logic to diverse domains. In the present Chapter, k-probing has been explicitly analyzed to determine its potential and novelty as a scheme for information retrieval, along with associated data structures, and to evaluate its performance in comparison with conventional hashing.

Three k-probing schemes (named K1, K2 and K3) are discussed. Finding the image under the k-association map of a key to be stored is analogous to hashing the key. Because of the homophonic nature of the map, several keys will have the same image, i.e., collide due to k-probing. This same homophonic property is used for collision management also. In K1 and K2 collision resolution is by open addressing and in K3 by chaining. Several examples are given to illustrate particular aspects of the schemes. K3 is the most general of the three schemes. Here space utilization, probe length and the complexity of constructing the k-association map are parameters under user control—the trade-off between these attributes has to be decided by the user. It is argued, and demonstrated with examples, that equiprobable maps in K3 yield space and probe-length performance comparable to well-designed hash functions. K-probing exploits the k-logic advantages of design simplicity, mnemonic value and synset generation. A combinatorial analysis of the three schemes is provided. The data structures visualized are hierarchical or tree-like, quite unlike the flat arrays used in hashing. While hash tables have to be periodically re-organized (because over a period collision resolution disturbs the hash table structure), there is no such requirement for these schemes because of the hierarchical data structures of k-probing.

1.10 Application of k-probing to mnemonic telephone numbers (Chapter 8)
A special-purpose application of k-probing is to generate mnemonic telephone numbers. Buttons on telephones are marked with letters too, with the intention that a
telephone number can be remembered using the corresponding letters, e.g., “GetATicket” is another denotation for the telephone number 4382842538. There are some limitations of this system. First, the mapping between digits and letters is static. For example, the numbers 5, 7 and 9 are encoded by the consonants jkl, pqr and wxyz, and there are no letters at all corresponding to the numbers 0 and 1. A telephone number with such digits will be denoted by a sequence of consonants and digits which are likely to have no or little mnemonic value. Second, since the mapping is homophonic, there will be several mnemonic word-sequences which will map into the same telephone number, only one of which can be assigned to a user.

In the proposed solution, a k-association map replaces the keypad. K-probing overcomes both the limitations stated above. First, the binding between numbers and letters is not static but depends on the mapkey (that determines the k-association map). Second, because a subset of letters in k-logic is superfluous and does not carry information, telephone numbers which would not readily yield mnemonic word-sequences will do so now. The cost of this solution is that the telephone handset (or soft phone) needs to contain a "k-interpreter" that converts a mnemonic string into a valid telephone number. (With telephone instruments containing embedded processors nowadays, this is a realistic and low-cost solution.)

For a given key, the string-to-number conversion has to be a perfect hash, because collisions are effectively disallowed, i.e., two strings cannot map to the same number. Therefore an algorithm is developed which, given a mapkey, constructs a perfect hash function interactively. This hash function is no other than the k-probe function that has been discussed in Chapter 7. Experimental results are given, with over 400 BSNL telephone numbers, a vocabulary of over 5000 words from a colloquial dictionary and mnemonic strings for the numbers generated by the algorithm. An analysis of the result identifies the causes for collisions due to the dialpad mapping and the manner in which k-probing resolves these causes. Typically the mnemonic string is about twice the length of the telephone number because the vowels are superfluous, whereas it is necessarily the same length with the dialpad mapping. Finally the time complexity of insertion, deletion and lookup of mnemonic strings in the k-probe function is briefly mentioned.

1. Overview
1.11 Conclusions and suggestions for further research (Chapter 9)

The final Chapter summarizes the main results obtained by applying the generalized katapayadi scheme to several domains of information retrieval and security. The significance of IMME is: especially when human expertise is rare and completely automated systems are not feasible, k-logic along with IMME yields satisfactory solutions without extensive human or computational effort.

K-logic is seen as a novel technique for homophonic substitutions. In particular, treating vowels as superfluous is a new idea which results in (1) enhancing the code-breaking strength of substitution ciphers and (2) organizing data in the form of hierarchical clusters that aid efficient recall.

The strength of k-logic in IMME is due to user-defined homophones. Being subjective, for an intruder they are as hard to break as the ones generated randomly. However, being map-based, their construction by machine is feasible. This feature enables user-controlled creation of homophones custom-made for the application at hand and simultaneously increases user productivity by assisting the human participant in the IMME application. The net result is that for all the applications that we have considered, our approach outperforms established techniques in a variety of ways. This claim pertains to steganography in particular, but holds for the other applications too.

Our approach to pronounceable password generation is based upon first scanning a dictionary to gather all combinations of pronounceable patterns of vowels and consonants, and then employing a personalized k-association map for the selection of a set of homophonic non-word strings to be used as a single person's passwords. This approach has multiple benefits. The passwords are user-friendly in the sense that the entropy, length and choice of letters are all user-specified. They are not disconnected and one-off, but share the same map. This feature imparts mnemonic value to the collection of passwords to be managed together. Superfluous vowels give rise to the novel concept of user-salted passwords that are resilient to social engineering attacks and also are useful in authorizing restricted access to system resources for guest users.

Analysis of the k-probing scheme for data organization shows that irrespective of the application domain, a systematic design of k-probing yields a novel search technique whose performance is comparable with that of a well-designed hash function. Besides
this it has the advantage of the usual katapayadi human-friendly features. For instance, our observation is that k-probing yields a richer set of mnemonic telephone numbers and hence gives greater satisfaction to subscribers.

The Chapter closes with the enumeration of possibilities for further research. They include: (1) developing a full-fledged IMME system using HCI theory that builds on the ideas discussed in the thesis, (2) developing a tutor interface using inputs from the disciplines of education and pedagogy and (3) further evolution of the models presented in the thesis wherein the human part of IMME is taken over by AI techniques.

Since the k-logic based models discussed here in assorted contexts depend on utilizing human intelligence with the help of machine assistance, they can neither be classified as deterministic nor stochastic. Our attempts have been aimed at convincing the reader about the validity of the research results using logical arguments, empirical testing and combinatorial analysis. Formal proofs of the experimental results and hypotheses presented here would require the exploration of interdisciplinary specialties in mathematics, linguistics, psychology and information technology.

1.12 Appendix: History and evolution of the katapayadi mnemonic

For the purposes of this research, it is sufficient to know merely that the original katapayadi scheme is no more than a one-to-many mapping from the 10 digits to the 33 consonants of the Sanskrit language, and furthermore, that in order to generate a mnemonic word sequence to denote a given integer, superfluous vowels or conjoint consonants can be added to the consonant string image of the number. In the present work, the katapayadi mapping is not restricted to its original Sanskrit context, but is applicable to any alphabet that can be partitioned into two sets that we may call consonants and vowels, respectively.

The origin of katapayadi, its place in medieval as well as current Sanskrit literature and literary forms and the scope for expression that it has provided to poets and creative writers—these topics are not relevant from the viewpoint of computing. Linguists and researchers have found katapayadi to be a fascinating subject and a secondary literature has emerged in recent years [Kak00, Raman97]. Readers whose primary interest is computing might be curious about katapayadi in its original form. To satisfy their curiosity, a few aspects of katapayadi have been compiled and placed

1. Overview

11
in an Appendix. The author has taken the liberty, in the absence of a detailed historical record, to speculate on the motivations of the creators and developers of this technique. It is hoped that the Appendix will provide interesting and fulfilling information that will stimulate the inquisitive reader.