International Journal of Electronic Engineering and Computer Science
Vol. 3, No. 2, 2018, pp. 22-28
http://www.aiscience.org/journal/ijeecs

Limitations of Turing Machine and Advanced Models of Computation

Naseer Ahmad[*], **Muhammad Waqas Boota**

Department of Computer Science, Virtual University of Pakistan, Lahore, Pakistan

Abstract

In this paper a discussion on limitations of Turing Machine and examine various models of computations is carried out. These new models are extending from Turing Machine called as SuperTuring model of computation. Also describe other Super Turing models of computation in a short. Dynamic interaction of internet, an infinite adaptation and robots are popular fields which cannot describe by Turing machine. We can do this with new models of computation as SuperTuring models of computation. Here, three popular SuperTuring modes of computation are Interaction machine, the π calculus and the $ calculus. In this paper it is explained that how these models solve the computational problems. In the future, SuperTuring models of computation will become the central programming paradigm.

Keywords

Entscheidungsproblem, Algorithm, Computation, Interaction Machine, The π Calculus, The $ Calculus

Received: May 6, 2018 / Accepted: June 15, 2018 / Published online: July 26, 2018

1. Introduction

Alan Turing in 1936 describes the assumption of Hilbert logical proof of all mathematical problems cannot be solved by computer and algorithms. Turing said that computers are too weak to solve all mathematical problems. However he introduced oracle machine and choice machine. These machines are very expressive than Turing machine in problem solving technique.

In the later, Turing machine used as a paradigmatic model of computation and Church-Turing Thesis used as a complete model for general problem solving and algorithms. There is a conviction that Turing introduce a universal model was inconsistent to solve all mathematical and computational problems. Turing's proof that Entscheidungsproblem is unsolvable and computational problems are unsolvable algorithmically by computers.

Gödel was first who proved the unsolvability of Entscheidungsproblem by using recursive functions and later Church using λ-calculus [4]. Recursive, lambda-definable and Turing computable methods said to be equivalent when these three classes of functions confirmed to effective function computation are properly defined. These equivalence methods of recursive, lambda-definable and Turing computable capture the effective computability of functions over integers.

The simplicity, elegance and power of Turing machines formed a paradigm of computation which destroyed earlier all computational and mathematical problems cannot be solved by algorithmically. In 1960 a new ACM computer science syllabus based on algorithms and Turing machines selected as a central role in computer science. In 1960 when first undergraduate computer science textbooks were written then Church-Turing thesis extended incorrectly to its strong form. When there is effective method for doing something with a computer then it will be computed by a Turing machine.

While the original Church-Turing thesis only applied to the effective computing functions over integers. This cannot

* Corresponding author
E-mail address: ms120400137@vu.edu.pk (N. Ahmad), ms120400080@vu.edu.pk (M. W. Boota)

extend the other types of computation such as functions over real and interactive computation. Church and Turing were known these limitations then Turing described other powerful models of computation. We exclude the all limitations of algorithmically computation in the other models of computation [5].

Now with development of new applications, computer science community believes that Turing machine and algorithm cannot provide a complete model for problem solving. Turing machine and algorithm cannot describe in the following areas of computer science like robots, internet and infinite adaptation from evolutionary computation. These areas of computer science require new models of computation or other models of computation. These new models of computation are able to describe in non-algorithmic computation as super Turing models.

In this paper we discuss the three super Turing models of computation refer to as Interaction machine, the π calculus and the $ calculus. We explain that how these models solve the computational problems than Turing machine. We expect that these new models of computation will become an active area of research. In this paper we encourage more research on new models of computation in order to solve the problems [2].

2. Related Work

Turing was born in 1912 and studied mathematics from Cambridge University in 1932. Turing extends the group theory models of Von Neumann. His paper "on computable numbers with an application to the Entscheidungsproblem" in 1936 proved that all mathematics problems cannot completely modeled by computers.

In 1940, he formed the computer model of German cipher code which helped the allies to win World War II. He formed the computer models of chess, human mind and artificially intelligence. These models suggested that computers can play chess better than humans before the end of the century. His paper was inability of Turing machine to solve mathematical problems. In 1960 computer scientists adopted Turing machine as a theoretical model to solve all problems of computing [1].

3. Tm and Entscheidungsproblem

Turing interested in Entscheidungsproblem, this is the most appealing conjectures in mathematics proposed by David Hilbert in 1918. Hilbert conjecture was any mathematical proposition decided by mechanistic. These logical methods disproved by Gödel in 1931. He expressed that any formal theory will be un-decidable theorems outside of its reach. Alonzo Church, Turing and Gödel are looking for alternate techniques in order to prove this undecidability result.

His paper "on computable numbers with an application to the Entscheidungsproblem" based on automatic machines (a machine). This can carry out any algorithmic computation. Turing described that these machines cannot compute all problems. Now he stated that the famous halting problem is undecidable.

The a-machine model consists of following:

1. One dimensional erasable tape of infinite length can store symbols and one on each cell of tape.
2. Read / write head tape head can move to left/right on the tape. This store one tape symbol at the current tape location.
3. Control mechanism can be in any bounded number of states.
4. Transition table have symbol under the tape head and the current state of the machine. This specifies the next action of the tape head and new state of the machine.

In the earlier, machine is in a special intial state and its control mechanism causes to read from current tape location. Now the control mechanism saw in the transition table that retrieved symbol on the current state of the machine. Its write a new symbol at the current tape location, transition to new state and moves left/right one cell on the tape. The computation terminates when machine reaches a special halting state.

Turing machine computation consists of the following:

1. Touring machine closed computation when all inputs are given in advance.
2. Touring machine is permitted to use an unbounded however only for finite amount of time and memory for its computation.
3. Each Turing machine computation starts similar initial configuration.

These characteristics are most suitable for modeling algorithmic computation. This prevents TM from modeling directly many aspects of modern computing systems [3].

4. Turing Machine and Algorithm

Algorithms and Turing machines are two basic concepts of computer science for problem solving. Turing machine tells the limits of computer science in the 1960s. Turing machine is a formal model of a computer which runs the particular program. Alan Turing invented Turing machine and introduced the automatic machine (a-machine) in 1936 paper. In his paper he shows the unsolvability of Hilbert's decision problem in mathematics to prove all mathematical problems.

The Turing machine is considered to model an arbitrary problem which solved by using mechanical methods. But, undecidable problem cannot solve by TM, while intractable problems can be solved and required too many steps. However, undecidable problems do not provide effective recipe are called non-algorithmic or non-recursive. There are exist an algorithms for intractable problems. But, it is run on deterministic TM which requires an exponential amount of time as a function of the Turing machine input.

There are number of various problems over any alphabet of more than one symbol is not countable, while, the number of all possible TMs is enumerable. Every TM shows that one problem. Hence it is clear that there exist problems which cannot be solved by TMs. It is clear that there exist problems, we cannot solve by TMs. In this way, Turing receive the result that unsolvability of the halting problem of the universal Turing machine shows all possible TMs. This keeps up correspondences to the unsolvability of Hilbert's Entscheidungsproblem.

An algorithm is one of the important concepts of current mathematics and computer science. It has roots in ancient Egypt, Babylon and Greece, introduced to mechanize computation problem solving. An algorithm should consist of a finite number of steps and well defined meaning. However, every algorithm represents in terms of a TM is now known as Church-Turing thesis. The TM model solved formally that there are some problems that cannot be solve by TM.

The TM model is too weak to solve the Internet, evolution and robotics problems. Its reason is that TM model is a closed model which requires that all inputs are given in advance. TM permitted to use an unbounded but only finite amount time and memory resources [16].

In the internet, web clients jump into the middle of interaction., this is done without the knowledge of server state. As we know that a dynamic set of inputs and outputs, parallel interaction of multiple clients and server formed a dynamic structure of internet communication and nodes. We require sequential and static full specification in the Touring machine.

Turing machine have problems like capturing evolution, as we know that evolutions and solutions are changed in every generation. The search for a solution is in general is an infinite process. It is true that whether search space is finite or infinite. The search for a solution in evolution computation is a probabilistic. This also requires infinite steps for search space.

Robots interact with environments which are more complicate and complex than robots themselves. Hence, it is clear that why usual computer science old fashioned AI. It is tried that to build a model of robots using Turing machines and algorithms [2].

5. Driving Home from Work

Here we discuss the problem of driving home from work (DHW). It cannot be solved by algorithmically and computation. Turing machine cannot solve computational problems, which contradicts the Strong Turing Thesis proving it is incorrect.

Here we consider a car we drive across town from work to home. The output should be a time series plot of signals which enable it to perform this task automatically. In the algorithmic point of view, we provide all inputs prior. The input of DHW includes a map of the city, the input needs to give the exact road conditions beside the way. In the static world, this input is specific but the real world is a dynamic. The presence of physical elements such as rain, wind and storms etc can effect directly and indirectly. So, it is doubtful that the prior inputs which we calculated are correct for DHW problem.

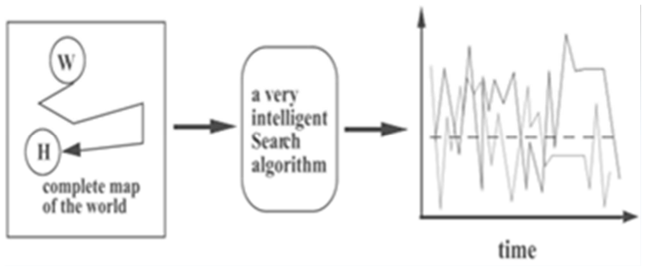
W
H
complete map of the world
a very intelligent Search algorithm
time

Figure 1. Driving home from work: the algorithmic scenario.

In order to avoid collisions, we should exact calculate the motions of everyone prior for computation. We suppose that human actions are computed ahead of time is an assertion of fatalism, so human beings cannot change this data because this data is come from computer scientist. Hence we conclude that the DHW problem is unsolvable.

In computational point of view, the agents of inputs are consist of a stream of images which produced by a video camera which placed on the moving car. The signals to the cars control are generated in the response of online images which steering off the road. So, the DHW example proves that we cannot solve this algorithmically and computationally.

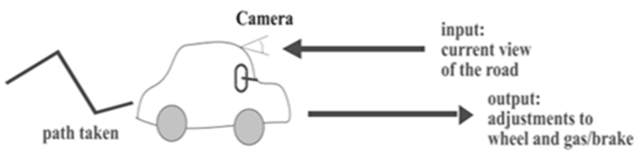
Camera
input: current view of the road
output: adjustments to wheel and gas/brake
path taken

Figure 2. Driving home from work: the interactive scenario.

So we reconstruct the inputs and change the computational model as well as the notion of computational problem. Algorithmic problems are computed off-line while interactive problems are computed on-line. So, the inputs and outputs for interactive computation is interdependent such as replace the video camera with a prerecorded video tape of the road. This will be equal to putting the blindfolds return to the driver [6].

6. Super Turing Models of Computation

Theoretical computer science developed from the several years of decades. The classical topics of computer science like automata theory, formal languages and computational complexity firmly established. But many areas like multi-agent interactive systems, robotics, client-server models and distributed database do not describe in the algorithms. The development and acceptance of computational models described in most expressive way than TM referred as Turing machine [17].

Super Turing computation mean any computation cannot be taken as a Turing machine and any other computation can be taken in TM. A majority of computer scientist accepted the concept of Church-Turing thesis that every algorithm described by a TM. But there is a problem by equating classical algorithms and Turing machine. Algorithms are well-defined and useful, but it is not possible to describe all computations by algorithm. This is not possible including Super Turing computation

In this paper we discuss three Super Turing models of computation which are more expressive than Turing machine and algorithms. Here we discuss also why these models are better than TM model particularly in the areas of internet, robotics and evolution.

6.1. Interaction Machines

Interaction machines introduced by Wegner in 1997, he described interaction of object oriented and distributed systems. The paradigm modify algorithm to interaction, the technology modified from mainframes to workstations and networks, and from procedure oriented to object-based and distributed programming.

As we already discuss that TM are too weak to describe interaction of object-oriented and distributed systems. Interaction machines suggested as a stronger model which captures better computational behavior for finite interactive computing agents [14]. The intuition which computing corresponds to formal computability, Turing machines break down the notion that computable is broadened to include interaction. While Church-Turing thesis valid only in narrow sense, when TM express the behaviour of algorithms. While broader declaration of algorithm precisely capture what computed is invalid.

Interaction machines enlarge the Turing machine model permitted by interaction like input and output actions determined by the environment at each step of computation. It also extends the closed TM model to an open system. But TM needs all inputs to show on the tape proceeding to the computation. During the process of computation, interaction machine permit inputs generated by dynamically and require inputs described by a potentially infinite stream. Hence any finite stream can extend by dynamically by interactive inputs. Interactive system interacts with external environments, but they cannot control. The behavior of interactive system is good to use infinite process which represents the cardinality of the real numbers than enumerable sequences. Interaction machine relate to TM by unbounded tape contents [15].

The number of inputs of TM is countable and number of potential streams of an interaction machine is non-enumerable. While IMs are not similar TM, we use interaction as a basic principle of their study which is a foreign notion for the closed world of TMs. As we know that TM knows in advance the contents of its tape, while the IM is sure that the input will receive the result of interaction. The evolution of bounded inputs and outputs form IMs different from TMs. Interaction machines are expressive due to unbounded tape contents, enumerable set of states, inputs and outputs. The set of inputs are dynamically bounded and assumed that other agents are non-computable. This leads to a more expressive model TM.

Interaction machine used to model the dynamic interaction for the internet. As we know that, they allowed only for bounded inputs, the number of states, inputs and outputs can be infinite. The canonical model of interaction machine called as Persistent Turing Machines. This is appropriate for evolutionary computation. Persistent Turing Machines have gene information from generation to generation.

Sensing and acting of robots are natural to model in IMs than TMs. Dynamically bounded sensory inputs are allowed to describe both uncertainty of sensors and actuators.

6.2. The п Calculus

Milner's π calculus is a mathematical model of processes, and they change interconnections when they interact. This is similar to IMs and built around the central notion of interaction. In this model, we actually are changing the connectivity of interactive systems. So, in this way the π calculus of mobile processes considered to be a dynamic extension of Milner's earlier model. While the majority of process like algebra and the π calculus use a synchronous message which is pass by handshaking. But the process of algebra consists of static communication channels. The π calculus allocate arbitrary link topology, we can change this by sending new links through existing ones. Processes reconfigure their interconnection structure when they execute and make it easy to model systems. When processes are moving between different locations and resources are allotted dynamically.

The π calculus is a basic model of computation both in narrower algorithmic and wider super Turing sense. Each basic model rest upon the small number of primitive notions, as we know that the π calculus have the primitive notions of interaction, just as TM have the notions of reading and writing a storage. Similarly, the recursive equations and the λ calculus rest upon mathematical function reduction. The π calculus includes the canonical approach to sequential computing. There is no definite measure on expressiveness of the π calculus. While, there is strong evidence that its primitives are necessary for a wide variety of purposes that include the encoding of functional and object oriented paradigm.

Here we explain that the π calculus is more expressive than TMs. It depends upon the interpretation of the π calculus replication operator. We can define the π calculus replication as the parallel composition of the process many times when needed. If the replication is unbounded or infinite then the π calculus is more expressive than TMs. It can model an infinite number of cells of cellular automata, and discrete neural networks. Neural network can be modeled in the π calculus as an infinite parallel composition of their related cells or neurons. Cells interact with the π calculus with message passing in and out operators. Without infinity, we suppose that the π calculus processes have the power of oracles, so we remain in the class of TM. We interpret that as many times as needed implies that the infinite number of replications. Hence such impetration of recursive definition, parallel composition and the π calculus is more expressive than TMs.

The π calculus provides the following support in the fields of internet, evolution and robotics.

1. We can model in a natural way interaction in the internet by message passing.
2. The π calculus allows the link mobility in communication topology.
3. We can model both asynchronous/synchronous, send/receive and point to point communication used to express unicasting, multicasting and broadcasting.
4. In the evolution, inherent parallelism allows in a natural way to describe the population of solutions. if not, then no support is provided.
5. Robots send/receive in communication primitives through dynamically created commutations channels.

6.3. The $ Calculus

The $ calculus is a mathematical model of processes which have both the final outcome of problems solving and interactive increment way. The $ calculus has a process algebra of bounded rational agents designed for interactive problem solving. This is introduced in the late 1990s. It is a formalization of resource bounded computation. Anytime algorithms are guaranteed which develops better result, when more resources are available [13].

The $ calculus have the primitive notion of cost in a similar way when the $ calculus built around a central concept of interaction. Interaction and cost ideas are very closed in the sense that cost confine the quality of an agent interaction with its environment. The $ calculus provides a support for problem solving by incrementally search for solutions. This also provides cost to direct its search. The $ calculus use the search method for problem solving is called kΩ-optimization. This is a general method which allows simulating many other search algorithms. This include A*, MINMAX, dynamic programming and evolutionary algorithms. This is appropriate to robotics, neural nets, software agents, and evolutionary computation. It is used for devise of cost languages, cost driven hardware, DNA-based computing, molecular biology and quantum computing.

The $calculus made its expressiveness from infinity of its operator without the magic of oracles. It is simple and easy to think that the implementation of unbounded concepts. Hence, implementation of scalable computers designed to think that a reasonable approximation of the implementation of infinity. Here we clearly know that it is completely unclear that to implement oracles.

The $ calculus has the same features as the π calculus for internet client server interaction. Mainly, it allows to model allocation of resources in order to optimize flow of information. Actually there is no problem for infinity of operators with scalability. The $ calculus describe in a natural way to state evolution. While, cost performance measure permits to model fitness function, as well as infinite search for global optima. In addition to, the $ calculus permits to optimize the quality of solutions and to reduce search cost.

While the behavior of robots, interaction with an environment and other robots can be defined in a uniform form with the $ calculus based Generic Behavior Message Passing Language (GBML). Uncertainty modeled by combing choice operator with costs using and fuzzy set membership function [2].

7. Other Super-Turing Models

In this section we discuss other Super-Turing models of computation.

7.1. C-machines, O-machines and U-machines

Choice machine, unorganized machine and oracle machines are super Turing models of computation which is defined by Turing. Choice machine and oracle machines drive their expressiveness from interaction principle. Unorganized machines drive their expressiveness from the infinity and evolution principle.

7.2. Cellular Automata

Cellular automata (CA) introduced by John von Neumann in search of models of self-reproduction, universal construction and universal computation. It consists of infinite of an infinite number of cells where every cell is a finite state machine. Cellular automata called as computation universal which refer their ability to implement any algorithm. Cellular automata can suggest ant Turing machine, but the reverse is not true [12].

A cellular automaton has great expressive power due to infinite principle. While infinite number TM tapes have a finite string at any one time. Cellular automata can construction universal which have the ability to construct arbitrary automata. Construction universality is one of the important field in artificial life and robotics.

7.3. Site and Internet Machines

Van Leeuwen introduced site and internet machines which capture the properties of computation. This is not modeled by Turing machine. Site machines are interactive Turing machines which are limited version Turing and Oracle. These are equipped with several inputs and output ports and communicate by sending / receiving messages. While, a site machine can exchange with the advice oracle in order to gain uncountable advice from it.

An internet machine models the internet. It consists of a finite but time varying set of site machines which work synchronously and communicate with exchanging messages. Every machine in the set is similar with its address and size of the set can grow at most polynomial by time.

Van Leeuwen proves that internet machine is not more communicative than a single site machine. But this proof sacrifices the dynamic nature of messages which assume that they are precomputed. Hence both internet and site machines are powerful than TMs.

7.4. Analog Neural Networks

Neural networks held computational cells similar to cellular automata. But the underlying network is not remaining homogenous, it can arbitrary disappear the transition functions which compute weighted sums of inputs from neighboring networks may be discrete.

Instead of having finite many cells, analog neural networks may compute non- recursive functions. They can compute the analog shift map is known as non- recursive. However, neural networks can be a standard model for super Turing analog computing and similar to the role of the TM in the Church-Turing thesis.

While the extra power of analog neural networks have their real valued weights which allows the neurons to take continuous values in their activation functions. Such a neural network may be idealized as chaotic physical system.

7.5. Evolutionary Turing Machines

Evolutionary Turing Machine means a series of TM. The outcome tape from one generation shaped the input tape to the next generation. ETM is similar to a Persistent Turing Machine or IM, because both hold a population of solutions and depiction of an evolutionary algorithm. The expressiveness of ETM is formed by evolving infinite populations, applying an infinite number of generations or applying variation operators which produce non -recursive solutions.

7.6. Accelerating Universal Turing Machines

In the end, Super Turing Model is the Accelerating Universal Turing Machine whose programs are executed at an ever – accelerating rate, allows infinite programs to finish time. AUTMs need the maximum of two units in order to execute any possible program [3].

8. Conclusion

In 1968, a new ACM computer science curriculum was base on algorithms, so TM chosen in order to complete a central role in computer science. As we know that, TM is very simple and elegant.

Here we forget that Entscheidungsproblem is unsolvable and mathematics cannot describe with algorithm relatively to create the foundations of computer science. Turing did not accept the TM as a complete model for computation.

TMs and algorithms are simple and elegance but incomplete, so theoreticians thought that TMs as a paradigm for complete model of computation. It reacts negatively against to go beyond TMs. Practically most of the models are developed which are equally expressive to TMs. TMs did not stretch the limits computational power for solving TM undecidable problems. New applications and requirements of a new class of problems are changing that situation. This is the clear opinion of Turing about new applications and problems.

Turing has very serious doubts about completeness of his model that's why he introduced three other models of computation: choice machines, oracle machines and unorganized machines. These models are more expressive than his classical model. Also Von Neumann and Ulam suggested the solution of universal constructability and using the help of more power powerful cellular automata. In the 1990s many other models are more expressive than TMs introduced. These models were distributed cellular automata, site and internet machines, analog neural networks, evolutionary Turing machines and accelerating universal Turing machines. The reader can read the complete discussion of SuperTuring models, where we discuss the three basic models underlying and unifying SuperTuring computation.

In this paper we discuss in detail three super Turing models of computation. These three models are more powerful than TMs. These three models capture dynamic aspects of interactive computation and designed for different purposes. Interaction machines designed to model dynamic aspects of computation which based on interaction. The π calculus developed around the central notion of interaction while its primary objective was to depicted concurrent systems with changeable communication topology. The $ calculus search best solutions under bounded resources.

Hope that this research is very simple and complete model of computation. in this paper a framework is provided to develop models going beyond TMs. Turing machine provide the foundations of theoretical computer science, programming languages and architectures. Hence super Turing models of computation will provide new foundations, hardware and software for computers.

References

[1] General structure of the Windows Mobile kernel http://msdn.microsoft.com/en-us/library/aa909237.aspx

[2] Overview of the Symbian architecture http://www.developer.nokia.com/Community/Wiki/Porting_iPhone_native_(Objective-C)_applications_to_S60_5th_Edition

[3] Overview of the iPhone OS architecture http://www.developer.nokia.com/Community/Wiki/Porting_iPhone_native_(Objective-C)_applications_to_S60_5th_Edition

[4] CFNetwork and other software layers on Mac OS X http://developer.apple.com/library/ios/#documentation/Networking/Conceptual/CFNetwork/Concepts/Concepts.html

[5] Overview of the Android architecture http://elinux.org/Android_Architecture

[6] The Blackberry MDS Runtime environment http://www.rim.com/symposium/press/pdf/BlackBerry_Mobile_Data_System_4.pdf

[7] A Survey of Mobile Platforms for Pervasive Computing By Hielko van der Hoorn. pp 16-36 http://www.cs.rug.nl/~aiellom/tesi/vanderHoorn.pdf

[8] Comprehensive Analysis of Smartphone OS Capabilities and Performance pp. 3-6 http://www.scf.usc.edu/~juihungc/project_page/doc/ee532_FinalReport.pdf

[9] http://en.wikipedia.org/wiki/IPhone

[10] Design a Mini-Operating System for Mobile Phone by Dhuha Albazaz pp. 1-2.

[11] Comparative Study of Different Mobile Operating Systems by T. N. Sharma, Mahender Kr. Beniwal, Arpita Sharma pp. 1

[12] Wegner P., Why Interaction is More Powerful Than Algorithms, Communica-tions of the ACM40: 5, May 1997, pp. 81-91.

[13] ACM Curriculum Committee (1968) Curriculum 68: Recommendations for academic programs in computer science. A Report of the ACM Curriculum Committee Science. Comm. ACM, 11, 151-169.

[14] Gödel, K. (1931) Uber formal unedtscheidare Satze der Principa Mathematica und Physik, 38, 173-198.

[15] Eugene Eberbach, Dina Goldin, and Peter Wegner "Turing Ideas and Models of Computation".

[16] Peter Wegner and Eugene Eberbach "New Models of Computation", 2003.

[17] Peter Wegner and Dina Goldin "Computation beyond Turing".