CHAPTER 2
LITERATURE SURVEY

2.1 Time Triggered Architecture

In continuation to introduction the discussion is further carried with a literature survey of Time Triggered Architecture, Basic Concept of Optimization Techniques, Importance of Travelling Salesman Problem (TSP) and Research Status in safety critical systems and by-wire implementable in aerospace, automobile, railway industries, medical and banking systems. The Composability and transparent implementation of fault tolerance are the main advantages of time triggered architectures. In the Time triggered architecture, system activities run on global synchronized time, which is quite contrast to event triggered systems.

2.1.1 Time Triggered vs Event Triggered

The advancements in computer and communication technologies predominantly, automotive and avionics systems have become more practical due to the extension of embedded computing systems with safety critical applications. In some cases they may contradict on application to application and the requirements may vary due to which there is no single model exists for system designs which have interaction in physical environment. Thus the design modeling depends on predictability or flexibility and resource adequacy or best effort usage strategies [22]. Hence selected design model can have tradeoff based on application.

In system domains like automotive or avionics, having the safety critical applications are adopted by x-by-wire components, if not can lead to catastrophe. The loss can be financial and/or involving lives, the magnitude of which can be more than the system cost. Hence these types of systems must be designed with resource sufficiency policy having fault tolerance and aggravate load handling capability. The two different approaches can be adopted in real time system design: *Event Triggered* and *Time Triggered.*

For system activities like sending a message or initiating any computational activity in event triggered architectures are started when the events occur in the environment, whereas, in *Time Triggered Architectures,* actions are initiated by the

progression of time. The general variation between Event Triggered and Time Triggered process depends on the control setting. Thus, the control becomes autonomous in Time triggered Systems and responds to the environment depending upon its schedule already predefined, however in Event Triggered Systems the control is not autonomous and response to the event happens as and when they encounter.

In the real time systems based on event trigger, the communication and the processing activities are started when there is a considerable change in the state is noted, i.e., the event occurred rather than the usual clock tick. Here, the well known interrupt mechanism is used for realizing signaling events. Hence, a clear scheduling process is described for event trigger based systems to realize the appropriate task that services the event [23]. The operating system of Event Triggered architecture is usually based on anticipatory scheduling and precise synchronization task execution. However, it is impossible to find out the maximum task execution time without taking into consideration of its possible interactions with other available tasks [24].

In an Event Triggered System, specific sensors have to be used to monitor the state changes. Such changes generate task activation requesting modification of connected objects. However, this task starts execution only when the scheduler has granted the processing time due to an event driven environment. With this effect, the internal object is updated properly only after the execution of task is completed. This change is reflected in the associated state. Consequently, this approach introduces temporal ambiguity; as a result it is not possible to identify exactly at what time the starting and ending of the task execution takes place. This is because, due to interruption or delay in task execution by higher priority tasks, or due to un-interruptible code segments or critical sections in the process. Temporal ambiguity may be introduced due to varying delay lengths. Further, temporal ambiguity may also be introduced due to clock skew. In some cases, decisions of time based scheduling can also lead to temporal ambiguity. The above explained temporal ambiguities influence system's expectedness and determinism, as it may not guarantee the appropriate environmental alterations in a well-timed manner, even though scheduling problem is solved [25].

The Time Triggered method is usually preferred in design of safety critical systems [26, 27]. Nowadays, in high end cars with x-by-wire design, most of the

communication needs are facilitated by time triggered architectures [28]. The time triggered architectures, used to provide characteristics required for hard real time systems, in addition it manages the complexities of fault tolerance and related dependable design model, which is a requisite for the systems with ultra high reliability [29]. Generally the failure rates around 10^{-9} failures/hour may be considered for ultra high reliable system. The periodic messages are transmitted at pre-determined points of time establishing membership information and allowing error detection. Redundancy is customary for such applications where the timing and any change in the function is not effective. Replica determinism is also supported in time triggered systems [30], which is a must to establish fault tolerance by inducing dynamic redundancy. Moreover, the Time Triggered systems maintain sequential composability [31] through an accurate specification of the interfaces used in the subsystems. However, the message transmission is decided independently by the communication controller of a Time Triggered Architecture. A temporal firewall created by the Communication Network Interface (CNI) of the subsystem, isolating the temporal conduct of the host and rest of the system.

In Time Triggered Systems, all the processing behaviors and communication related activities are initiated at preset points of time. The periodic clock interrupt, is the only interrupt used in the system, which partitions the band of time into sequences of evenly spaced small time granules [25]. The system flexibility and improved resource utilization are the main benefits of using event triggered system, whereas in time triggered approach predictability is the main benefit. In Time Triggered Architecture, the activation of a task to update an object based on suitable state changes of related states and this happens at preset points of time. As a result, Time Triggered Architecture is better suited for reliable systems than Event Triggered System. The task activation in time triggered approach has a pre-determined commencement period, thus it can be scheduled off-line and this can be without the prior information of upcoming events. Thus for time triggered architecture predetermined task activation period might be defined, whereas for event triggered architectures the minimum period for arrival of activated task has to be assessed. A most important limitation of timed triggered approach is that, the aggravated state of affairs has to be known a priori in the system development process.

In an Event Triggered System, if nodes want to communicate and willing to send information on the bus, it may do so unless it is not engaged by another node. Collisions may occur in this approach when more than one node tries to access the bus and send messages. CSMA/CD (Carrier Sense Multiple Access with Collision Detection) is one of the methods used to resolve the problem, where the node that is trying to access the bus for communication with other node, listens on the bus for possible collision of its message. If possible collision is detected; the node is made to wait for a prescribed amount of time then it retries in sending its message. If several nodes need to use the bus at a time, the waiting time may be long and much of the bus capacity is wasted on tackling colliding messages. A variant called CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) is used to address this type of problem. Here CSMA/CA tries to involve either a type of bus access ordering method or to have distinct prioritization of the nodes. All nodes have to listen to the bus every time for access and further succeeding node otherwise highest priority node is allowed to use the bus for message transfer. The above said principle is adopted in many systems to avoid the collisions. As discussed above, in a priority based system, to identify the highest priority, some kind of procedure is required to determine which node is given permission out of those that are attempting for bus access. This course of action is called the Arbitration.

The Event Triggered Architecture implementation is quite simple and easy for expansion. When new nodes are added to the bus, the existing nodes need not be reconfigured. The major drawback of the Event Triggered Architecture is its non-determinism. Hence, there is no guarantee that when a message transmission will be successful [32]. In a Time Triggered System, only one node is permitted for transfer of data at any instant of time due to this the risk of collisions are eliminated on the bus. Thus, the time at which a particular message is to be transferred can be guaranteed and the predictable behavior as required for systems with high reliability can be achieved for example in control-by-wire applications. Further, strictly periodical behavior is needed in some control systems, which can be implementable with the above approach. For such an approach, all nodes of the network are required to have a common view of the system time, which is applicable only by Time Division Multiple Access (TDMA) mechanism.

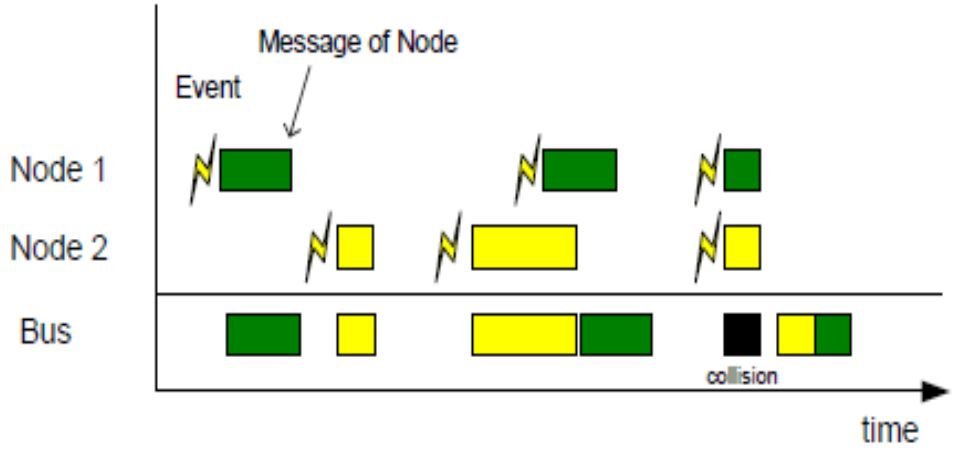

Figure 2.1 Collision of Event Triggered messages on a bus.

The figure 2.1 above illustrates the collision of event triggered messages on the bus when accessing at the same time. Clearly it is seen event triggered message collision occurs and for time triggered message this can be avoided as shown in figure 2.2 below. As every system has its own advantages and disadvantages, time triggered systems too has one disadvantage, if a node has not sent anything in its prescribed time window, the window will stay as unused. Further, the Time Triggered systems need to be well synchronized and expansion becomes a complicated affair. Here, nodes to be added in future should be included in the time schedule right from the beginning of design, or else the network may have to undergo extensive reconfigurations and rescheduling [33].

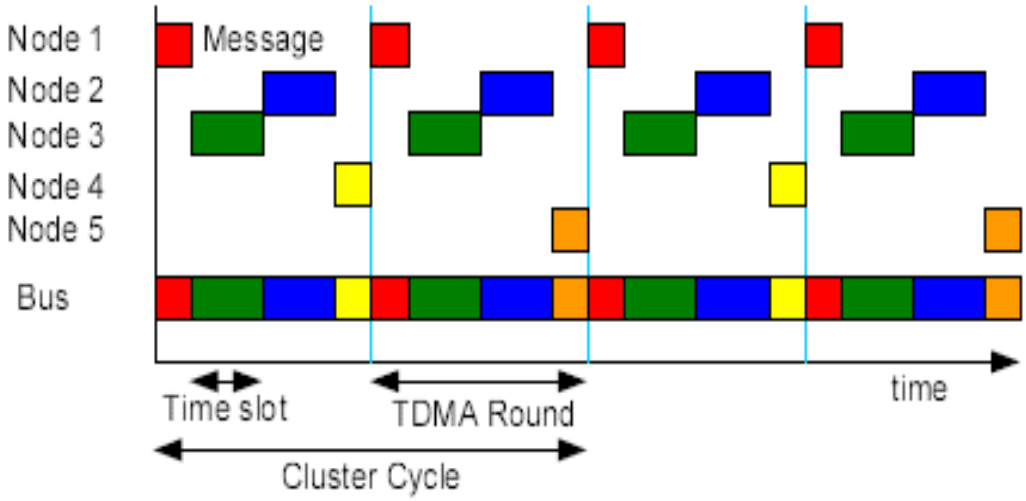

Figure 2.2 Message transfer in Time Triggered Environment

However, in the systems where safety is not so important parameter, like in soft real time systems, the traditional event triggered control approach is considered because of its flexibility advantage and also due to efficient resource utilization. Event Triggered architectures maintain methods similar to dynamic behavior resource allocation method and also resource sharing. The provision of resources in event triggered systems can be influenced towards the demands, as a consequence allowing timing failures to take place at some point in worst-case scenarios for better solutions. It is known that the relationship involving the resource usage is recognized and resources involved can be networked which can also be multiplexed depending on the condition that the guarantee of communication latencies and also providing sufficient buffering capacities.

The classification of distributed systems with real time operations either Event Triggered or Time Triggered systems depending on the control approach and can also be classified as *federated* and *integrated systems* depending upon level of integration. These system classes vary as per the allocation of functions to the system under consideration. In the federated system, all the major subsystems are dedicated systems working independently but these subsystems are interconnected by gateways with a limited level of coordination realized in between with a common goal. On the other hand, an integrated system is a single distributed system mapped on to several subsystems; such a frame work is used for the development of an *integrated architecture.* A federated system strengthened by fault isolation and also inducing error containment, an integrated architecture can be a better coordinating system, significantly reduces hardware which leads to improvement in the dependability [29].

The following Table 2.1 illustrates the comparison between event triggered system (CAN protocol) and time triggered system (TTP/C protocol). So that we can clearly prefer the time triggered approach for designing hard real time system in particular to safety critical application.

Table 2.1 Event Triggered Vs Time Triggered

EVENT TRIGGERED	TIME TRIGGERED
Event messages	State message
Temporal control derived from the occurrence *Unpredictable*	Temporal control derived from the sequence of time *Predictable*
Flexibility	Interoperability
Large jitter	Minimal jitter
No precise temporal specification of interfaces	Precise temporal specification of interfaces
Good for sporadic data	Good for regular data
Membership difficult	Membership easy
Probabilistic access	Replica determinism

2.1.2 The Time Triggered Protocol

The Time Triggered Protocols (TTPs) are designed for the hard real time systems [34]. The TTP/C is the first protocol that achieves low cost and high dependability at a time. This makes it suitable for high speed data transfer, with a tolerance of single failure system for safety critical applications such as avionics, railways and automobile manufacture industry.

The TTP assumes that underlying communication network to be highly reliable. A typical TTP system is a cluster having Fault Tolerant Units (FTUs), and each of which consists of more than one node, interconnected by a communication network with dual communication buses [35] as shown in figure 2.3 below.

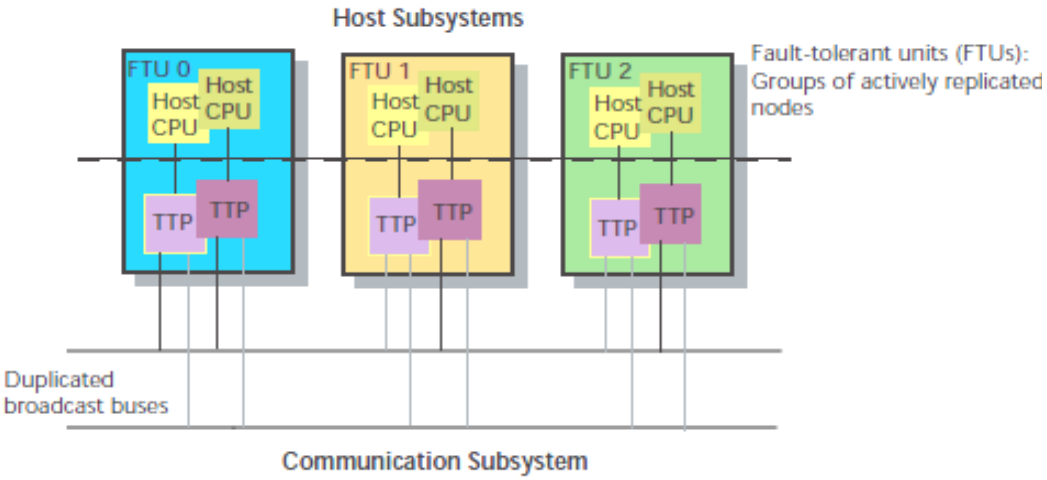

Figure 2.3 Fault tolerant units in a cluster

The host and the communication controller are taken as two subsystems of the node as shown in figure 2.4 below, with Communication Network Interface acts as interface between them. The communication controller consists of local memory having the MEssage Descriptor list (MEDL) that determines at what point of time the transfer of message from the node and acceptance of the message from the other nodes should take place. It also contains independent hardware devices like sensors and actuators along with the Bus Guardians. The bus guardians monitor the controller access from the replicated buses. The bus guardian terminates the controller operation if it detects variations or timing violations during the process [36].

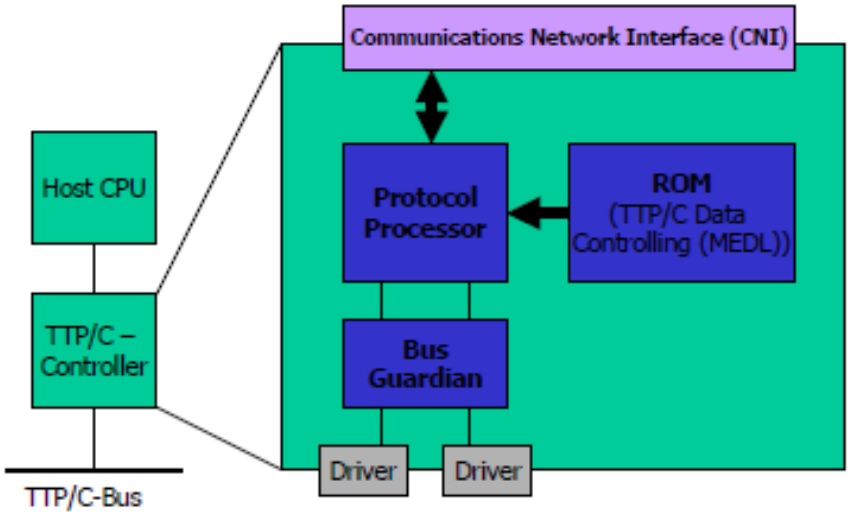

Figure 2.4 A typical TTP node

The TTP is a Time Division Multiple Access (TDMA) protocol where each node can send its message individually on the shared replicated communication buses in a specific assigned time slot in a predetermined TDMA schedule, which is shared by all the nodes. The TTP/C protocol can be applied on different types of physical media. In the OSI model it is described above the physical layer i.e, providing error free communication across the network, but it supports both electrical as well as optical physical media underneath.

The time triggered nature of TTP comes with the following properties:

- *Use of apriori Knowledge*: TTP has a sparse time base [36] in which events only happen at pre-determined time points in the globally synchronized time base. Since it is TDMA based, TTP allows every message to be broadcasted and uses a simplified acknowledgement scheme [37]. The synchronous schedule is generated at the initialization phase of the system.
- *Clock Synchronization*: TTP maintains Byzantine resilient clock synchronization with a precision about 10^{-6} second by performing the Fault-Tolerant Average (FTA) algorithm periodically, with the support of hardware in most cases [36].
- *Composability*: The Communication Network Interface (CNI) in between the communication controller and the host, allows the host to be built and tested autonomously. Since the CNI will remains same during the system integration, the properties of the subsystems will not be invalidated after the system integration. This enables the adoption of a cost effective bottom-up design method in developing time triggered hard real time systems.
- *Fault Tolerance*: The independent bus guardians ensure that a node either it is delivering the message at the correct assigned time or not, i.e. fail silent operation is supported by the nodes. The host software will act as authoritative to achieve fail silence in value domain by ensuring space redundancy and/or time redundancy that is transmission of erroneous or malicious messages that are not detectable are ensured to be stopped. This is done by detecting all the internal failures of the host as such to stop its operation [38]. The TTP also provides *replica determinism*: that is a message has to arrive at all the

recipients exactly at the same time with the same contents or else it will not arrive at all [37].

- *Membership*: The membership service in TTP makes a facility to inform all nodes in the system about the failure of a node with minimum delay. Each node broadcasts its membership vector together with any message it sends. Each node updates its membership vector after receiving a message from another node by checking the status of the corresponding field in the message.

The TTP/C protocol is the full version of the TTP designed to provide all services required for highly dependable systems with fault tolerance and are distributed hard real time system sufficient for safety critical applications [36] and a new variant of TTP is available now, the scaled-down version named TTP/A aiming at economical and easy integration of smart sensor networks, which share a compatible CNI and supports modular design with easy management of transducers.

2.1.3 System Architecture

The system architecture is developed over time triggered paradigm systems which is fully synchronous and built around time triggered network used for message transmission and also for synchronization [38]. The below figure 2.5(a) shows four Electronic modules (ECUs or nodes) connected to a common bus. Each electronic module consists of a communication controller which disassociates the communication subsystem with the host subsystem. The communication process between the nodes is governed by time triggered protocol TTP/C [39]. The nodes of cluster exchange its messages in between by TTP/C communication protocol. According to the static schedule, the communication subsystem decides message transmission and reception of a particular node is relevant or not.

The TTP/C network consists of two channels, replicating each other. The host subsystem of Electronic module executes the application, may possess an activator or a sensor. The communication subsystem in the node is communication controller designed to execute the features of TTP/C protocol. The nodes can be interchangeable in case of faults, hence specified as Smallest Replaceable Unit (SRU). As said earlier the Fault Tolerant Unit (FTU) is the combination of more than two nodes as shown in figure 2.5 below. The node and the network is called the cluster [40,41,42]. A

particular service is delivered by the Fault Tolerant Unit even if a node fails from the group.

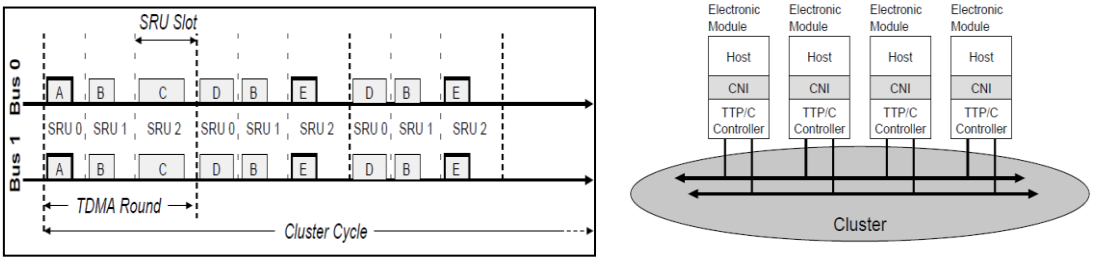

Figure 2.5 (a) TTA scheme with Electronic modules connected by replicated communication channels (b) Cluster Cycle Representation.

The bus accessing scheme is derived from the global time notion controlled by cyclic Time Division Multiple Access, where every active module has its own TDMA slot. Each node in its allocated TDMA slot sends or receives the message. The sequence of TDMA slots which form a TDMA round is shown in Figure 2.5(b). Every TDMA round carries the messages to the nodes in the allotted slots. After completion of a TDMA round, another TDMA round is started but maybe with a different or in sequence message in the same pattern as the previous one. The number of TDMA rounds varies depending upon the messages in for transfer and this determines the length of the cluster cycle. When the cluster ends, the transmission pattern starts again in new cluster cycle. The system does two types of schedules; which are communication schedule and scheduling of individual node. All the nodes in the system share same network for message transmission and reception which is basically of broadcast type. As different tasks are performed by the node, the node scheduling is different and it should be framed optimistically. The operating system executes this schedule in time triggered manner.

The propagation of errors due to malfunctioning of the node or collection of malicious data from the other nodes is prevented by communication network interface i.e, babbling idiot problem can be prevented. A temporal firewall is created by the communication network interface of the ECU. Any single hardware failure is tolerated and if any malicious host produces erroneous data can never interfere in

correct operation of the cluster. That is fail silence condition is guaranteed by the communication controller of the node. Thus the system becomes fault tolerant and applicable to safety critical systems like automotives, avionics, railways etc, [1].

2.2. Basic Concept of Optimization Techniques

Optimization refers to obtaining the best possible solution for a problem by giving group of limitations (or constraints). The Optimization related problems are very common in most of the real time disciplines and domains. In optimization related problems, we have to obtain optimal or near-optimal solutions with respect to some preferred goals. More often than not, we cannot solve problems in one go, but have to follow some method which directs us through for problem solving. Most often, the process of obtaining solution is broken into number of steps which are executed one by one. Commonly the steps are recognized and define the problem, construct and solve the prescribed model, and evaluate and implement the solutions [43].

A simple block diagram presented in figure 2.6 below enables us to understand the optimization process easily. This flow process explicates the role of Optimization methods or algorithms in real time problems to achieve the best optimal solution. In simple words it can be said that Optimization algorithms or procedures applied for real systems are like adding a grain of salt to enhance the performance not only just in the form of outputs, but finding the best optimal solutions in minimum span of time. Here the response at processing stage i.e. for mathematical model is concerned as very important attribute, as it is the one which leads to betterment in the preferred algorithm or method.

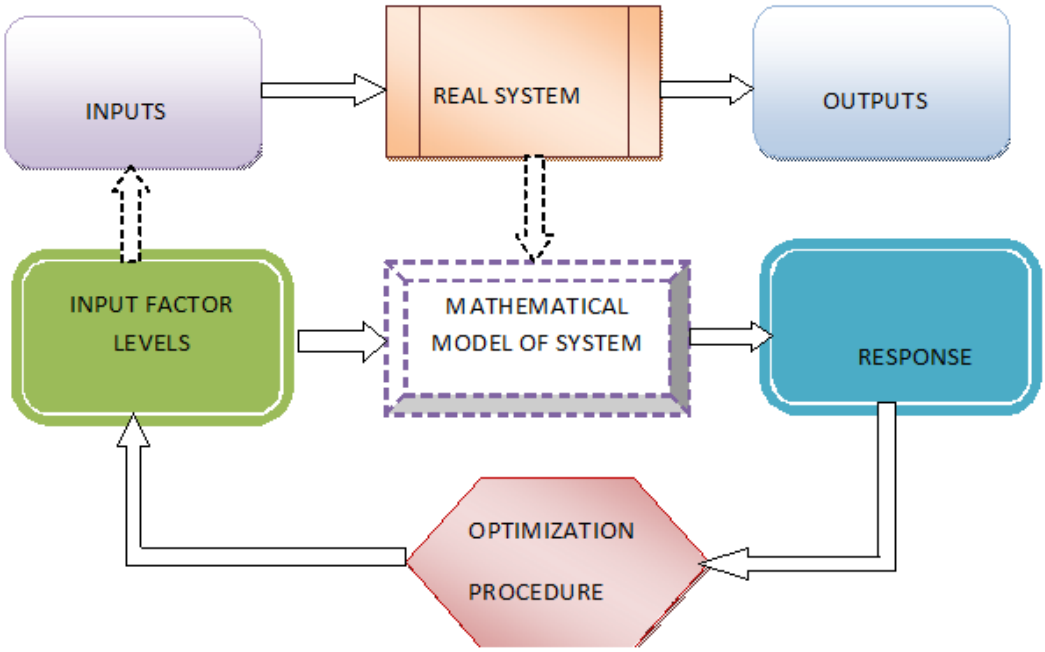

Fig 2.6 Optimization Process flow

Optimization procedures are inherent with various parameters; the default classification is given in figure 2.7 below. The algorithms are employed based on their problem type or variable of significance. Once again the sub-classification arises depending on the requirement or application considered i.e. constraints and dependency.

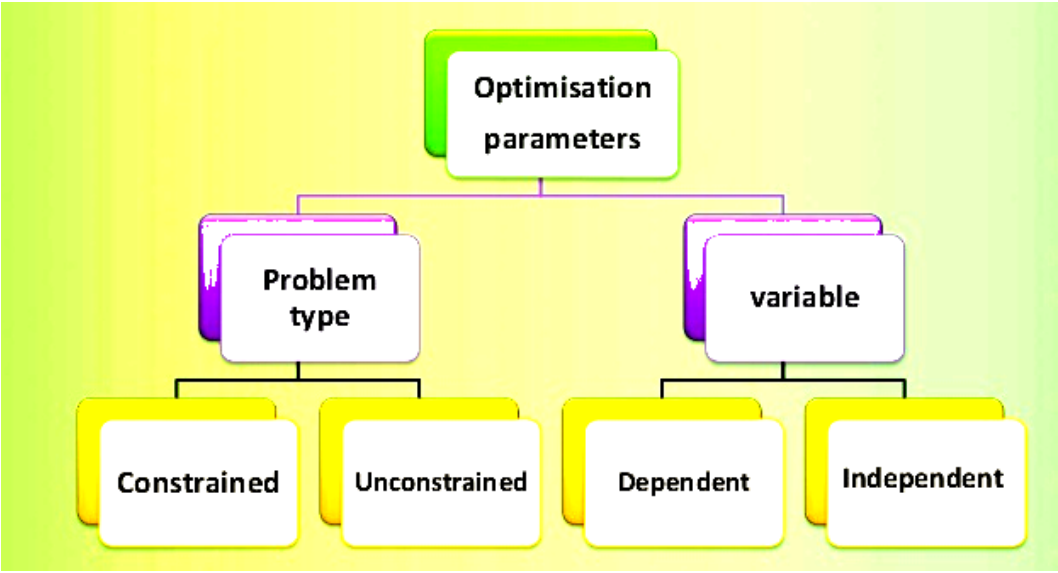

Fig 2.7 Optimization Parameters

There are number of *general purpose optimization heuristics* as given in [45] used for system level or subsystem level design optimization, such as *simulated annealing* [46-49], *tabu search* [50-52], and *genetic algorithms* [53-56]. Many researchers or designers either acclimatize with heuristics approach or go for greedy based customized design algorithms and some approaches make ways to come out of local optima. Firstly, the design optimization heuristics were employed at system level which simplifies hardware/software partitioning of a problem i.e, codesign of a real time system is mapped on a uniprocessor system, by partitioning some functions may be implemented on ASICs or FPGAs and rest on the system processor for accelerating the system to much higher level [56-59]. Further, more sophisticated or advance methods were considered for the design of complex embedded systems [60-64]. Then, these methods were applied to the design of distributed heterogeneous embedded systems [65]. For example in [66] and [67], multi-cluster distributed system is designed. More recently fault tolerant systems with optimization related design issues have received much attention from the research and development community.

2.3 Importance of Travelling Salesman Problem (TSP)

In this work the travelling salesman problem is used which is the backbone for this research to get optimized solution. This is an optimization problem which is used to acquire shortest path to journey through the given number of cities [68]. Travelling salesman problem states that given a number of cities and with their distances, the traveler is to go through all the cities only once and back to the city from where he started and minimizing the cost of travel. In simple words it can be explained as a salesperson has to make a visit of all places with a condition that he has to go to a place only once and come back to his origin such as to calculate how much time has been spent to visit each place. This path is described as the tour and the path length is prescribed as the cost of the path.

2.3.1 Origin of TSP

The Traveling Salesman Problem was extensively studied in the 18th century by an Irish mathematician Sir William R. Hamilton and the British mathematician Thomas P. Kirkman. The book named 'Graph Theory' authored by Biggs et.al, explains the work of Hamilton & Kirkman in a clear and understandable way giving a good justification of their effort [69]. From literature it is believed that the general form of the TSP has been first deliberated by Kalr Menger. Later on this problem was extensively promoted by Hassler, Whitney & Merrill. A comprehensive account of work about the connection between Menger & Whitney's work, and the use of TSP can be found in Schrijver's work [70]. The main objective of the use of this approach has been to reduce the cost of travel and the best possible way to visit all the given cities, exactly once. In mathematical sense, Given the number of cities n to be visited exactly once, the total number of possible routes covering all cities can be given as a (n-1)!/2 set of possible solutions.

2.3.2 Review of TSP

A comprehensive review of TSP related work has been carried out in International and National arena by different authors is presented in this section.

Varshika Dwivedi et. al. [71] in their work solved the problem using Genetic Algorithm operators by utilizing TSP approach. To improve the solution space the genetic algorithm approach is used. The crossover and mutation are the important stages in execution of the genetic algorithm, they proposed a new crossover operator called Sequential Constructive Crossover (SCX) operator. The SCX operator is useful for selection of best edges of the parent's structure and creates new offspring. The SCX operator is compared with existing crossover operators and said to be having better quality solution. They also compared their work with Greedy method, and Dynamic programming method along with Genetic Algorithm to solve Travelling Salesman Problem.

Naveen kumar et.al. [72] has carried out survey on application of the TSP with different genetic algorithm operators such as selection, crossover and mutation techniques were used in the work to solve the problem.

In a research work by Omar et.al [73] proposed to improved genetic algorithm where the new crossover operator called Swapped Inverted Crossover (SIC) for producing better tours. Further population reformulation operation is carried out without modifying its fitness function. Instead of single mutation they carried out multi mutation operation by selecting individuals repeatedly several times wherein each individual of these are mutated using adjacent neighbors. They further processed with partial local optimal value with mutation operator which selects randomly the sub-tour of individuals to obtain better solution. Then the method finds the tour that generates the local minima by this sub-tour and exchanged with the original sub-tour. They also used rearrangement operation to obtain the maximum cost among the all adjacent cities on the tour. The final aim of this research is that all the operations used together produces the better result.

In a paper presented by Chetan Chudasama et. al. [74] travelling salesman problem is also solved by using basic genetic algorithm operators. This research basically brings a relative study of basic selection methods like Roulette Wheel, Elitism and Tournament selection methods for optimizing TSP tour to find best possible solution. The old school method called Roulette wheel selection method which is famous for its interesting gaming qualities, selects the best possible available individual based on proportionality to its fitness value. However in Elitism selection process the individuals are selected based on their fitness values, while in tournament selection method, randomly every individual is paired with another in the population. In the paper they said that when the population size is small all the three selection methods give similar results but when it is large enough better results are obtained by Elitism method.

Similar to the above work Kanchan Rani et.al [75] concentrated on selection operator and first applied Roulette Wheel method and then used a selection method known as Stochastic Universal Selection (SUS) method in which N equally spaced pointers are used to select the parents. After comparison between the two it is found that the SUS method gives superior results for the condition that population size should be small but as the population size increases Roulette Wheel gives improved results. They also carried out work with another Genetic Algorithm modified crossover operator called Order Crossover (OX). This crossover method makes two

cut points of the space and it is selected. The nodes present in between the two cut points are copied and remaining nodes are selected other parents in relative order excluding the existing nodes. They observed from the obtained results that this method gives the better results compared to existing crossover methods. In concluding remarks they also referred that the Elitism method may also give a good result when the population size increases.

Md. Lutful Islam et.al. [76] proposed a parallelizing Hadoop Map/Reduce framework to solve the travelling salesman problem by using genetic algorithm process. In their proposed work the crossover methods that were used are firstly Order Crossover, secondly a Two Point Crossover operator and lastly Partially Matched Crossover operator. The Genetic algorithm for the TSP problem is parallelized to produce the best possible solution in an acceptable execution time. The reduced execution time of algorithm is due to the Map/Reduce framework in which the Mapper and the Reducer functions run concurrently on different nodes. Even the problem size increases this framework can be adopted. The Hadoop Distributed File System (HDFS) can be used for the larger data storage. In brief the authors also explained about Hadoop and Map/ Reduce framework and their applications.

A crossover method called sequential Constructive Crossover method is proposed and adopted by A. Arananayakgi et.al. [77] in their work to solve travelling salesman problem using genetic algorithm operators basically to reduce the cost. The main objective of the work is to get high quality solutions in reasonable time and this is possible the by producing fitness function by use of selection, crossover and mutation operators during the process. The process selects the better edges from chromosomes of parents and generates new offspring (may or may not contain same edges as the parent chromosomes). As a result a binary matrix representation of chromosomes is also presented in their work.

To obtain optimum solutions to travelling Salesman Problem, B F A P Merz [78] presented an approach which is a combination of local search heuristics and generating algorithms. This approach uses local search techniques to find local optima in a given search space. The local optimum is searched in the space in order to find the global optimum using genetic algorithm. The new genetic operators realized by

the above technique can improve the quality and competence of the solutions for a set of symmetric and asymmetric TSP instances.

Mohammad Al Kassassbeh et.al. [79] stated a simple and fast method to obtain the solution where Shared Crossover method is used, which mainly aims to reduce the execution time. Here the potential shared paths between cities are passed to the next possible generation. Once the application of crossover is done, it is ensured that none of the cities reappear in child chromosomes. This process also results in reduction of execution time.

In literature survey carried out by Zakir H. Ahmed et. al. [80] on crossover methods to solve travelling salesman problem based scheduling. In the comparative analysis presented with various crossover methods, to choose an efficient one for implementation of Genetic algorithm over travelling salesman problem. They compared with Sequential Constructive Crossover (SCX), Generalized N-point Crossover (GNX) and Edge Recombination Crossover (ERX) in their experimentation and found that the sequential constructive crossover gives the high quality results.

Saloni Gupta et.al. [81] also solved the Travelling Salesman Problem using genetic algorithm. In their work the Euclidian distance between each city is entered in matrix format and randomly initial population is generated. As usual the crossover and mutation operators are applied in the process. Here, a two point crossover is used the genes are replaced with each other in the chromosomes. This process is continuous and terminates when the condition is satisfied.

But one of the first heuristic techniques addressing TSP problem was proposed by Russell and et.al [82]. The proposed algorithm was an extended version of the Lin & Kernighan heuristic [83]. Another heuristic based on an exchange procedure for the mTSP was suggested by Potvin et al. [84]. a parallel processing approach was adopted by Fogel [85] to solve the mTSP using evolutionary programming. The above researchers considered 25 and 50 cities to solve and get optimal solution using the evolutionary approach with TSP.

2.4 Research Status in the concerned area

In the recent past several comparative studies have come up for multiprocessor task scheduling problem. Many heuristics & Meta-heuristics have been developed by research community in this field to have better solutions for multiprocessor task scheduling. Naturally in Parallel processing systems scheduling process encounters many multi-objective problems. Usually the Scheduling problems are dynamic and possibly based on incomplete data. If the scheduling to be static, the project should be complete and the changes are observed only after schedule plans are announced. This depends on the duration of the project and the objectives of the multiprocessor system. Due to unanticipated disturbances or incomplete data and also possible poor estimation may result in variations in dynamics. Thus, deriving an optimal schedule is time and again mystified along with meeting the existing constraints. And also adapting new constraints may lead to changes in the problem structure. The Scheduling problems have many constraints and they appear in many forms: *temporal constraints* such as "the office is opened only on Mondays, Wednesdays and Saturdays"; *precedence constraints* such as "The interfacing programmes can only be written when Analog to Digital and Digital to Analog converters are connected on the development board"; *availability constraints* such as "two workers are available in the field for morning session and five workers are available in the afternoon session"; and *combinations* such as "The programmer is available on Monday morning session to write interfacing programmes". The added Constraints can turn a fairly uniform solution space with many optima into a non-uniform space with some possible feasible solutions. Generally a plan may have many bottlenecks with little bit of flexibility for changes and also they may have parts that are almost unconstrained.

Normally, the scheduling problems are used to be NP-hard, means there are no known algorithms for finding optimal solutions in polynomial time. The existing algorithms are used to solve some form of the problem, but they may take a long time (i.e. more than polynomial time), if constraints are added or the problem size is increased. Therefore, much of the research work is carried on either to simplify the scheduling problem to prepare the algorithm for finding a better solution, or designing an efficient heuristics algorithm to find better solution. Sometimes, the problem relates to finding a feasible solution only, and it is not sure that a feasible solution is

always obtained. Many researchers have developed and designed solutions and have been proposed for implementation. In early methods exactly solved problems finding simplified versions were realized, but researchers found in reality problems to be handled are too large and leads to complications for achieving any exact solution. For example, decision tree structures were used to specify every possible alternative. With them heuristic methods were designed either to find simply feasible solutions, or to find good solutions for the real time difficult problems. Most of the research now is concentrated in designing better heuristics for scheduling problems related to particular requirements. However, heuristic solutions are dependent on a specific set of constraints or problem formulation, and devising novel heuristic method is difficult at best.

Chunlin Li et.al. [87] proposed that Greedy algorithm can be used for dynamic heterogeneous resource environment through a homogeneous communication scheduling environment. Generally, the Greedy approach is one selected to solve problems like the job scheduling. According to the greedy approach -“A greedy algorithm always makes the choice that looks best at that moment. That is, it makes a locally optimal choice in the hope that this choice will lead to a globally optimal solution" [88]. Many variants have come up in greedy algorithms and the authors proposed a variant called Deadline Constrained Based greedy approach to improve the task completion time. Here, the greedy algorithm used is to minimize the turnaround task of individual tasks, ensuing overall improvement in completion time.

Boctor et.al in his work used simulated annealing process to search the combinatorial space of sequence permutations. The annealing process generates a sequence of tasks over which the schedule is created by heuristics. This method found to be similar to branch & bound technique. The Branch & bound technique has the limitation that it depends on the size of decision tree, but this process can be applied to a bigger problem. In simulated annealing the cooling process is the critical parameter and also the selection of neighbourhood operator plays a major role. Boctor applied simulated annealing approach to Patterson problems [89] and stated that he obtained fairly good response. Boctor concentrated on precedence feasibility in which precedence feasibility tasks are swapped, restriction applied to neighbourhood

operator. The annealer generates precedence feasibility list, which is used by heuristic scheduler for generating resource feasible schedules.

Stefka et al [90] adopted Ant Colony optimization Algorithm in their work for designing the scheduling algorithm mainly concentrating on high throughput for distributed systems. In these types of applications achieving best fitness solution is mandatory. In their work the system communication with remote user and the local user was asynchronous. Whereas Graham Ritchie et al [91] also proposed ant colony optimization algorithm in their work but combined local and tabu search technique to find shorter schedules. This was tested with benchmark problems rather than on other techniques found in literature. It conveys that methods are least bothered when compared to attaining the required objectives.

The ants colony system was developed by Marco Dorigo based on the idea of ant movements which aimed to search for an optimal path in a graph [92]. The movement of the ants is always dependant on the obstacles, how to overcome them and shortest path to reach the food by taking the help of fellow ants. Thus he solved the travelling salesman problem by introducing artificial ants. In this algorithm the past history is used to update the pheromone value. The pheromone value is the basis for finding the shortest path.

Holland derived the most popular genetic algorithms [93] for scheduling problems. The algorithms apply evolutionary scheme allowing fast exploration of the search space of schedules, and also allows to quickly finding good solutions. Further, the algorithm permits the designed scheduler to be employed to wide ranging problems [94].

In the work by Kim et al. [95] deterministic scheduling problem was considered in which multiple jobs with s-precedence relations are processed and this is done on multiple identical parallel machines. The main objective of this method is to minimize the overall completion time. The s-precedence relation employed here is between the two jobs *i* and *j*, the job *j* is inhibited from processing until the job *i* commences the processing. This is quite different from the standard definition of precedence relation in which the job *j* should not start until job *i* completes.

In the work by Hwang et al. [96], the challenges of task scheduling in multiprocessor systems were addressed. Here, the parallel programs are represented as

directed acyclic task graph (DAG). These DAGs were designed to be executed on multiprocessors bearing some communication overhead costs. For solving this problem they used Genetic Algorithm and designed a new encoding mechanism. The mechanism uses a multifunctional chromosome that uses the precedence representation called the priority based multi chromosome (PMC).

An efficient method for multiprocessor scheduling based on genetic algorithm was also developed by Hou et, al. [97]. The crossover operator was developed with task graphs having dependencies overlooking the communication delays in the process. The result obtained by their method is within 10% of the optimal schedules compared to others.

Wu et al. [98] proposed a new method based on Genetic Algorithm in which both valid and invalid individuals in the population are allowed for processing. This modified Genetic Algorithm processes with incremental fitness function. A satisfactory solution is obtained by gradually increasing the difficulty of fitness values. This method does not find applicability if the problems are scaled up, such as multiple tasks in an application. This is due to the time spent on evaluating invalid individuals which may not become valid ones.

A new modified genetic algorithm is proposed by Kamaljit Kaur et.al [99]. This new Genetic algorithm is based on heuristics adopted for scheduling static tasks onto distributed identical parallel system. The overall completion time is minimized by this algorithm and also the throughput of the system is increased.

Another novel Genetic Algorithm is proposed by Kwok et.al [100] with an aim to improve and schedule with high performance, adopted with scalability, and fast running times. L.D Davis [101] has worked with the basic principles of genetic algorithms amid variations in the population size, with different initialization methods, in defining fitness function, variation in selection process and also experimented with replacement policy in crossover and mutation processes. Also discussed about the issues related to implementation by one of the most popular meta heuristics.

A little research has been seen to design optimize the Time Triggered systems within the perspective of fault tolerance. As seen in Pinello et al. [102] a simple heuristic is designed by combining several static schedules in order to mask fault

patterns. Passive replication is used in [103] to handle a single failure in multiprocessor systems so that timing constraints are satisfied. Multiple failures are addressed with active replication in [104] in order to guarantee a required level of fault tolerance and satisfy time constraints.

Ceyda Oguj et al. [105] concentrated on flow shop scheduling problem in their work and proposed tasks scheduling by genetic algorithm on multiprocessor system. The main inspiration behind their proposal is that most of the present day systems are multitasking and they require much attention to avoid the confusions in obtaining the best optimal results. Thus they developed a new crossover operator (NXO) and compared it with already available PMX crossover with comparative analysis. A number of initial tests were carried out for regulation of main parameters of GA such as population size, crossover rate and mutation rate.

Correa et al. [106] proposed a modified GA by the use of list heuristics in the crossover and mutation in a pure genetic algorithm. In this methodology the selection operator is of least concern and hence neglected. This method is alleged to dramatically improve the quality of the solutions that can be obtained with both a pure genetic algorithm and a pure list approach. Unfortunately, the shortcoming of this method is that the running or computation time is much larger than when running the pure genetic algorithm. As minimum computation time is the main constraint that has to be maintained to achieve the optimum results, this approach has been not considered by the future researchers thereafter.

M. R. Bonyadi et.al [107] proposed Bipartite Genetic Algorithm (BGA) for reducing the maximum computation time for task scheduling on multiprocessor system, such that optimal results are obtained in minimum time. They carried out an initial test to lay down the constraints of Genetic Algorithm for enhanced performance and then compared with Genetic Algorithm based as well as other heuristic based algorithms in terms of Standard Deviation, average make span, best obtained make span and iterations.

A hybrid evolutionary algorithm was presented by Goh et al.[108] for task scheduling on heterogeneous multiprocessor environment. Here the GA operators were modified with respect to the multitasking application. The results achieved in their proposed work verified that the genetic operators proposed perform better, when

combined with the local search operators than when any one of the operators is omitted.

The multi-objective task scheduling problem was addressed by P. Chitra et al. [109] for heterogeneous distributed computing systems (HDCS) with two objectives of makespan and the reliability index. In this approach reliability was considered as the chief parameter that has to been given more attention as makespan has been already analysed by many researchers before this method. Two Multi-Objective Evolutionary Algorithms were developed to solve the multitasking problems and experimental investigations were carried out on a range of random task graphs and a real-time numerical application graph. The results obtained have shown that, Multi-Objective Evolutionary Algorithms are compatible for achieving good pareto optimal solutions in a single run for a wide range of task scheduling problems.

A modified list scheduling heuristic (MLSH) was developed by M. R. Mohamed et.al [110] and also a hybrid model was designed for task scheduling in multiprocessor system which is a combinatorial process of Genetic Algorithm and MLSH. The three diverse depictions for the chromosomes of Genetic Algorithm were projected in this procedure and they are (i) the task list, (ii) the processor list and (iii) combination of the both the lists. The proposed Optimization procedure here does better than the other procedures in terms of best makespan, average makespan and the efficiency of the processor. Accordingly, quite a lot of techniques have been well studied till at this moment to solve the multitasking problem based on Genetic Algorithms. However, for the most part of these methods consider the task scheduling problem on single criterion based multiprocessor systems, which may fail to obtain optimal results if multiple constraints are considered. In the above work, the task scheduling problem with minimization of the bi-criteria multiprocessor system has been considered.