

Perceptual Evolution for Software Project Cost Estimation using Ant Colony System

Nikhata Akhtar

M.Tech, B.Tech (Computer Science & Engineering)
Assistant Professor, Department of Computer Science & Engineering
Integral University, Lucknow, India

ABSTRACT

My proposed work is inspired by the experiment that uses expert judgment for estimation of the cost on the basis of previous project results. In this paper estimator can use Analogical strategies as well as Algorithmic Strategies as they wish. The proposed method is divided into two phases. First phase computed the probability of each selected factors by ant colony system. Second phase combines the value of these factors to calculate the cost overhead for the project by using Bayesian belief network. Once this overhead is computed productivity is directly calculated which can be converted in effort and cost. Our computation gives the Cost Overhead that depends on various factors. Till date Ant Colony Optimization Algorithm has provided solutions for the problems that have multiple solution and user are interested in best solution. This algorithm provides a proper heuristic for the problem and computes the best possible solution. It gives the solutions in terms of probability, i.e. The most likely occurred solution and the best solution. It was first introduced in Travelling Salesman Problem for finding the minimum cost path. We have mapped our problem in a simple graph by using a questionnaire. That gives the minimum length path, the path that obtains minimum deviation from the nominal project for each factor and their encouraging results from proposed technique.

Keywords

Cost Estimation, Bayesian network, Ant Colony, Algorithmic-Estimation Strategy, Optimization, Swarm Intelligence.

1. INTRODUCTION

Today scenario the software engineering cost estimation plays an important role in software engineering practice, often determining the success or failure of contract negotiation and project execution. Cost estimation's deliverables, such as effort, schedule, and staff requirements are valuable pieces of information for project formation and execution. They are used as key inputs for project bidding and proposal, budget and staff allocation, project planning, progress monitoring and control, etc. Unreasonable and unreliable estimates are a major cause of project failure, which is evidenced by a CompTIA survey of 1,000 IT respondents in 2007, finding that two of the three most-cited causes of IT-project failure is concerned with unrealistic resource estimation [1].

In the last three decades, many software estimation models and methods have been proposed and used, such as COCOMO, SLIM, SEER-SEM, and Price-S. Software cost estimation is considered to be more difficult than cost estimation in other industries. This is mainly because software organizations typically develop new products as opposed to fabricating the same product over and over again. Moreover, software development is a human-based activity with extreme uncertainties from the outset. This leads to many difficulties

in cost estimation, especially in early project phases. These difficulties are related to a variety of practical issues.

The software engineering cost (and schedule) models and estimation techniques are used to play a number of motives. These include budgeting, tradeoff and risk analysis, Project planning and control, software improvement investment analysis. In the today scenario Swarm intelligence is a relatively new approach to problem solving that takes inspiration from the social behaviors of insects and of other animals. In this paper we are using the Ant Colony Optimization Algorithm is a relatively recent approach to solving optimization problems by simulating the behavior of real ant colonies [2]. The Ant Colony System (ACS) models the behavior of ants, which are known to be able to discover the shortest path from their nest to a food source. Although individual ants move in a quasi-random fashion, performing relatively simple tasks, the whole colony of ants can collectively accomplish sophisticated movement patterns. Ants accomplish this by depositing a substance called a pheromone as they move. This chemical notch can be detected by other ants, which are probably more likely to follow a path rich in pheromone [3]. This notch information can be utilized to adapt to sudden unexpected changes to the terrain, such as when an obstruction blocks a previously used part of the path shows figure 1.

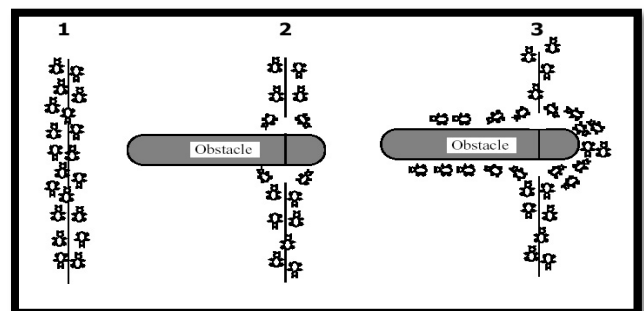


Figure 1. The Ants moving between the nest and a food source are blocked by an obstacle

The shortest path around such an obstacle will be probabilistically handpicked just as frequently as a long path however the pheromone notch will be more quickly reconstituted along the shortest path, as there are more ants follows this way per time unit shows figure 2. Since the ants are more inclined to handpick a path with higher pheromone levels, the ants rapidly converge on the stronger pheromone notch, and thus deflect more and more ants along the shorter path. This particular behavior of ant colonies has inspired the [4] Ant Colony Optimization algorithm, in which a set of artificial ants co-operate to find solutions to a given

optimization problem by depositing pheromone notch throughout the search space [5].

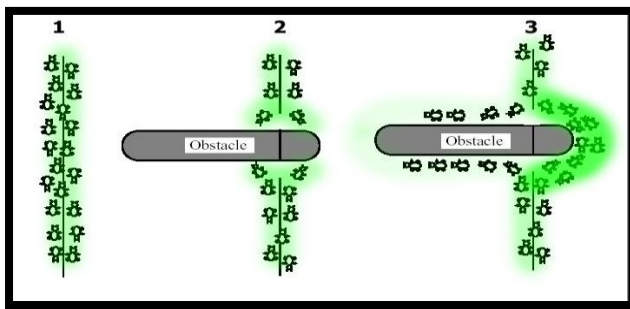


Figure 2. The Pheromone build-up allows ants to reestablish the shortest path.

2. THE OPTIMIZATION OF ANT COLONY

The Swarm Intelligence (SI) is a property demonstrates by some mobile systems such as social insect colonies and other animal societies that have collective behavior [2]. Individuals of those systems such as ants, termites and wasps are not generally considered to be individually intelligent however they do demonstrate a degree of intelligence, as a group, by interacting with each other and with their environment. These systems generally consist of some individuals sensing and acting through a common environment to genesis a complex global behavior. They are sharing many appealing and promising features that can be sucked up to solve hard problems. Furthermore, they are particularly well suited for distributed optimization, in which the system can be explicitly formulated in terms of computational agents [6]. The swarm encourages methods in computational intelligence areas is the Ant Colony Optimization (ACO) method. This method is encouraged by the foraging behavior of an ant system and has many well-turned applications to discrete optimization problems.

2.1 The Ant Colony System

The Ant Colony System algorithm is a The Ant Colony System algorithm is an example of an Ant Colony Optimization method from the field of Swarm Intelligence, Meta heuristics and Computational Intelligence. Swarm intelligence research originates from the work into the simulation of the emergence of collective intelligent behaviors of real ants. Ants are able to find surpassing solutions to the shortest path problems between the nest and a food source by laying down, on their way back from the food source, a notch of magnetize substance a pheromone. Ant Colony Optimization (ACO) is an instance in designing Meta heuristic algorithms for combinatorial optimization problems.

The trait of ACO algorithms is the combination of an antecedent information about the structure of a promising solution to a posteriori information about the structure of previously obtained surpassing solutions. The term Meta heuristic [7] derives from the composition of two Greek words. Heuristic derives from the verb *heuriskein* which means “to find”, while the suffix Meta means “beyond, in an upper level”. A met heuristic is a high-level strategy which guides other heuristics to search for solutions in a possibly large set of problem domains shown in figure 3. A met heuristic can be seen as a general algorithmic framework which can be applied to different optimization problems with

relatively few times. So far ACO has been applied with surpassing prosperity to a number of problems and scenarios, ranging from classical traveling salesman problems to a [8] variety of scheduling problems from constraint satisfaction problems to dynamic vehicle routing problems from routing in wired networks to routing in wireless mobile ad hoc networks, from data mining , facility layout, etc.

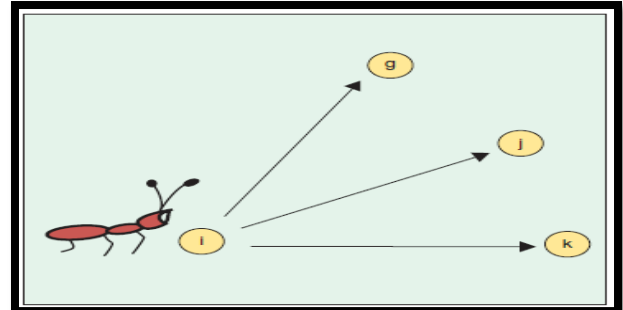


Figure 3. An ant in the city I choose the next city to visit via stochastic mechanism

3. THE SOFTWARE COST ESTIMATION

The Software cost estimation is the process of predicting the effort required to develop software systems. Many estimation models have been proposed over the last 30 years. Basically models may be classified into 2 major categories firstly algorithmic and secondly non-algorithmic. Each has its own strengths and weaknesses. A key factor in selecting a cost estimation model is the accuracy of its estimates. Unfortunately, despite the large body of experience with estimation models, the accuracy of these models is not satisfactory [9]. A software cost estimating model, like any other computer-based model, “is a ‘garbage in - garbage out’ device: if we put indigent sizing and attribute-rating data in on one side, you will receive indigent cost estimates out . Accurate cost estimation can help to classify and prioritize development projects with respect to an overall business plan. It can be used to determine what modality to commit to the project and how well this modality will be used. I have proposed a model for estimation that is based on Ant Colony Optimization.

In this paper the basic idea behind ACO algorithms is to simulate the foraging behavior of a swarm of real ants using a swarm of artificial ants working as cooperative agents to construct high quality solutions using a construction procedure. In recent years, software has become the most expensive component of computer system projects. The bulk of the cost of software development is due to the human effort, and most cost estimation methods focus on this aspect and give estimates in terms of person-months.

4. PROBLEM ASSOCIATED WITH EXISTING SOFTWARE COST ESTIMATION TECHNIQUE

In the real world accurate software cost estimates are critical to both developers and customers. They can be used for generating requests for proposals, contract negotiations, and scheduling, monitoring and control. Underestimating the costs may result in management approving proposed systems that then exceed their budgets, with underdeveloped functions and indigent quality, and failure to complete on time.

Overestimating may result in too many resources committed to the project, or during contract bidding, result in not winning the contract, which can lead to loss of jobs. In Algorithmic cost modeling a model is developed using historic cost information that relates some software metric (usually its size) to the project cost. An estimate is made of that metric and the model predicts the effort required. It uses function point which is more useful in MIS domain and problematic in the real time software domain. Expert Judgment Method involve consulting with software cost estimation expert or a group of the experts to use their experience and understanding of the proposed project to arrive at an estimate of its cost [10].

This method cannot be quantified. It is hard to document the factors used by the experts or experts-group. The expert may be some biased, optimistic, and pessimistic, even though they have been decreased by the group consensus. Estimating by Analogy means comparing the proposed project to previously completed a similar project where the project development information is known. Actual data from the completed projects are extrapolated to estimate the proposed project. The choice of variables must be restricted to information that is available at the point that the prediction required one has to derive an estimate for the new project by using known effort values from the analogous projects. Possibilities include means and weighted means which will give more influence to the closest analogies. The Putnam (SLIM) Top-down is estimating method is also called Macro Model [11]. An overall cost estimation for the project is derived from the global properties of the software project, and then the project is partitioned into various low-level components. SLIM is based on Putnam's analysis of. In SLIM, Productivity is used to link the [12] basic Rayleigh manpower distribution model of the software development characteristics of size and technology factors. It often does not identify difficult low-level problems that are likely to escalate costs and sometime tends to overlook low-level components.

It provides no detailed basis for justifying decisions or estimates. In Bottom-up Estimating Method the cost of each software component is estimated and then combines the results to arrive at an estimated cost of the overall project. It aims at constructing the estimate of a system from the knowledge accumulated about [13] the small software components and their interactions. The leading method using this approach is COCOMO's detailed model. It may overlook many of the system-level costs associated with software development. It may be inaccurate because the necessary information may not be available in the early phase. It tends to be more time-consuming. It may not be feasible when either time or personnel are limited. A checkpoint is a knowledge-based software project estimating tool from Software Productivity Research (SPR) developed from Capers Jones' studies [Jones 1997].

It has a proprietary database of about 8000 software projects and it focuses on four areas that need to be managed to improve software quality and productivity. It uses Function Points (or Feature Points) [Albrecht 1979; Symons 1991] as its primary input of size predicts effort at four levels of granularity: project, phase, activity, and task. Since a function point is believed to be more useful in the MIS domain and problematic [14] in the real-time software domain. Do not depend on a single cost or schedule estimate. Use several estimating techniques or cost models, compare the results, and determine the reasons for any large variations document the assumptions made when making the estimates.

5. PROPOSED SOFTWARE COST ESTIMATION MODEL

In this paper my work is inspired by the below experiment that uses expert judgment for estimation of the cost on the basis of previous project results.

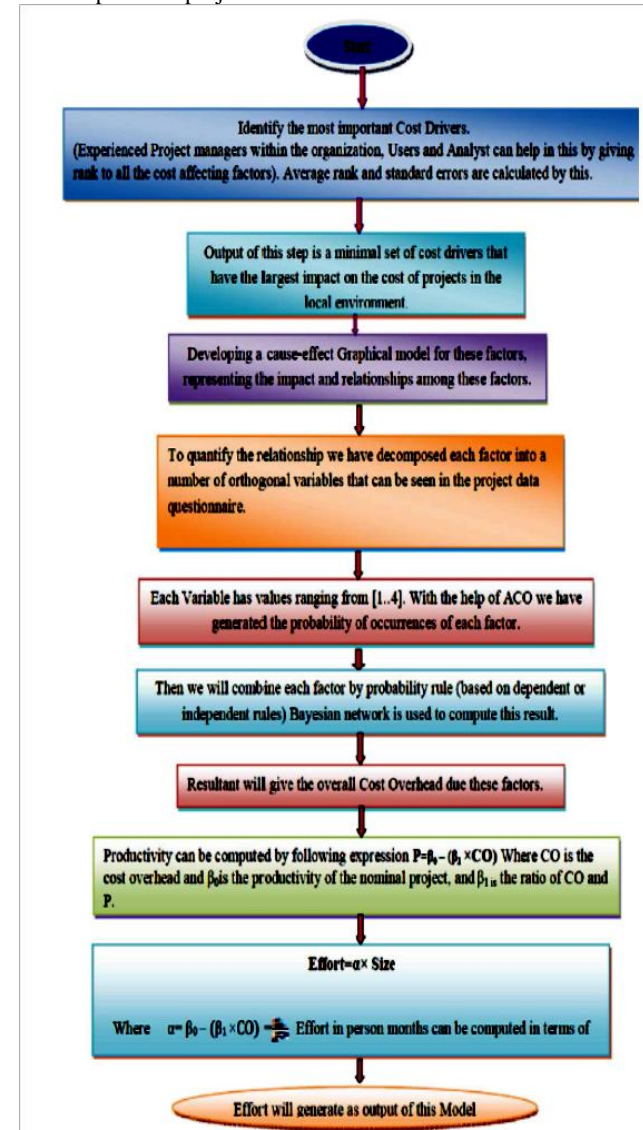


Figure 4. The Proposed cost estimation model

The estimator can use Analogical strategies as well as Algorithmic Strategies as they wish. The proposed method is divided into two phases. First phase computed the probability of each selected factors by ant colony system. Second phase combines the value of these factors to calculate the cost overhead for the project by using Bayesian belief network. Once this overhead is computed productivity is directly calculated which can be converted in effort and cost shows figure 4. The cost overhead is defined as an additional percentage with the cost at the top of the cost of a project run under optimal condition. The project only hypothetically runs under optimal condition, this type of project is called nominal project. The overhead cost represents the difference between the actual cost and the nominal cost of the projects. We can say that overhead is directly proportional to the deviation from nominal cost to the actual cost of the project. First the nominal cost of the project is decided then overhead factor are identified. To identify the most important cost drivers we choose the experienced project managers, user and analyst

related to organization for giving rank to all the cost affecting factors. They can also comment on the completeness of each cost driver. As a result the average rank and standard deviation for all cost drivers are generated. The output of this step is a minimal set of cost drivers that have the largest impact on the cost of projects in the local environment.

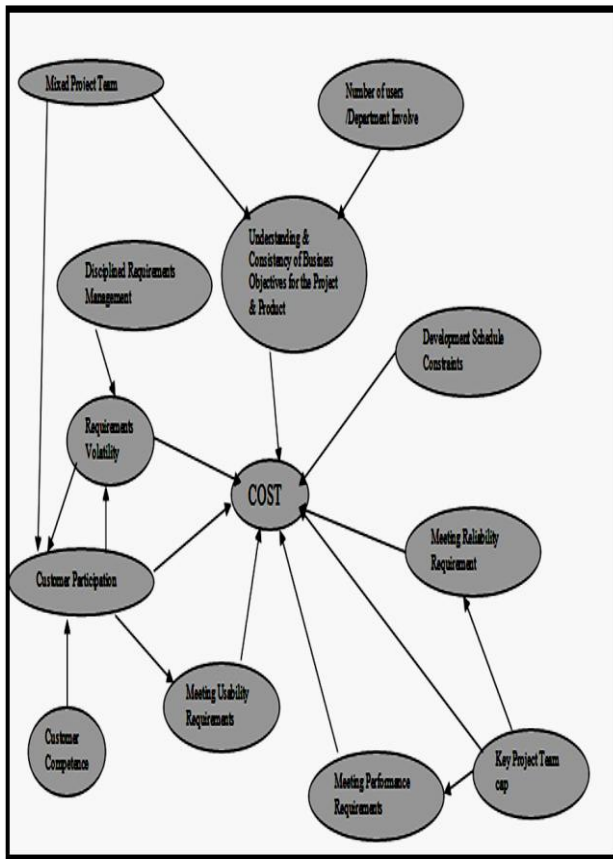


Figure 5. The cause-effect Graphical model for these factors

Once the minimal set of cost drivers has selected the next step is to develop the cause-effect relationship among them. The cause-effect graph model of these 12 important factors and it also represents relation with the factor whether they have direct or interaction relation. We consider whether they have a negative or positive impact on cost or each other. In figure 5 gives the relationship within the factors and cost, but this gives only cause and effect detail. So to quantify these factors, each factor is decomposed into a number of orthogonal variables that measure that factor through a questionnaire. For this purpose we have generated a project data questionnaire and a survey is performed for this questionnaire, in the survey expert person from the industries gives the value from 1 to 4 to each variable. After calculating the value of each variable of a specified factor we calculate the overall impact of that factor on the cost. Each factor is considered as a cost overhead multiplier. To compute the best result most likely occurred result for each individual factor ant colony system (ACS) plays a vital role.

5.1 Combining Impact Factor Using Bayesian Belief Network

In this paper using Bayesian belief network will be very helpful for this purpose so it is based on the cause effect relationship. For our calculation the relationship is presented

in figure 5. Each edge represents a relationship where the incoming arrow represents that receiving node is the effect of sending node. For calculation Bayesian belief network is based on a bays' theorem which is states as follows. The conditional probability of an event is the probability obtained with the additional information that some other event has already occurred. The term $P(B/A)$ represents the conditional probability of event B when event A has already occurred.

$$P(B/A) = \frac{P(A \cap B)}{P(A)}$$

When we are dealing with sequential events, we can use Bays' theorem, where new additional information is obtained for a subsequent event, and that new information is used to update the probability of the initial event. The probability of event A, given that event B has subsequently occurred, is

$$P(A/B) = \frac{P(A).P(B/A)}{[P(A).P(B/A)] + [P(A').P(B/A')]}$$

By using above equation we can combine each interrelated events, for our computation we can generate the overall cost overhead after combining the each dependent and independent factors. This will give us the complete information about overall overheads related to the projects. Expert easily predicts the effect drivers in terms of pert of cost drivers in terms of percentage. They give the deviation from nominal to actual projects. In standard economic terms, productivity is the ratio between the amount of goods or services produced and the labor or expense that goes into producing them [9]. The assumption that follows, then, is that software productivity is the ratio between the amount of software produced to the labor and expense of producing it Productivity is computed by following expressions.

$$P = \beta_0 - (\beta_1 \times CO)$$

Where CO is the Cost Overhead and β_0 is the productivity of the nominal project and β_1 is the ratio between CO and P. As we know productivity is inversely proportional to cost overhead. Effort computation can be performed by following expressions.

$$\text{Effort} = \alpha \times \text{size}$$

Where $\alpha = \frac{1}{P}$.

After computation of effort cost of the project can easily be computed. Productivity in terms of Output (KLOC) per person-month can adequately capture both cost and schedule concerns. If productivity is higher, it should be clear that the cost in terms of person-months will be lower. (The same work can now be done with fewer person-months.

5.2 The Ant Colony System Computation

In this paper size and effort data were collected retroactively for recently completed projects. These projects were considered to be representative of the types of projects that are conducted within the organization. All of the projects that we collected data on were considered to be “successful”, that is they had been completed, were fully operational, and were deemed to be admissible quality. Although these criteria did not lead [15] to the elimination of any project in our case study, such a selection is in general necessary in order to use a baseline of comparable projects, with consistent and meaningful size measurement. Size was measured in terms of non-comment lines of code excluding code produced by code generators. In addition, project managers filled up the questionnaire for their respective projects in order to obtain the data to feed our cost overhead model. In this dissertation we have computed the probability of impact of each cost affecting factor by using Ant Colony System [16]. We have to validate that the Ant colony system algorithm provides better and more nearly accurate results. For this purpose we have used a project data questionnaire that is sent to many experienced industrialists for their opinion on each cost affecting factor. Then we have combined the results from each person and this algorithm gives more refined results in after each iteration.

$$p_k(r,s) = \begin{cases} \frac{[\tau(r,s)] \cdot [\eta(r,s)] \cdot \beta}{\sum_{u \in M_k} [\tau(r,u)] \cdot [\eta(r,u)] \cdot \beta} & \text{If } s \notin M_k \\ 0 & \text{If } s \in M_k \end{cases}$$

- (1)
- In the given expression $p_k(r,s)$ is the probability of the ant k moving from node r to s .
 - $\tau(r,s)$ is the amount of pheromone contained on edge r to s . Initially it is set to 1.
 - $\eta(r,s)$ is the heuristic value of the edge (r,s) that is inversely proportional to the length of the path, for my computation it is given by the expression: $(\text{max-length})^{-1}$ where max is the maximum length of the path.
 - β is the attractiveness of the path in terms of ants on that particular edge.
 - u is the neighbor nodes of r .

- M_k is the set of nodes visited from r .

Local trail Update

- $\tau(r,s) \leftarrow (1-\alpha) \times \tau(r,s) + \alpha(\tau)$ (2)
- Where α is evaporation constant set to 0.2

Global trail Update

- $\tau(r,s) \leftarrow (1-\alpha) \times \tau(r,s) + \alpha \left(\sum (1-\alpha) \times \tau(r,s) \right)$ (3)
- $\delta \tau_k(r,s) = \begin{cases} \frac{1}{CT_k} & \text{if node selected} \\ 0 & \text{otherwise} \end{cases}$
- This is used only for the best path.

6. EXPERIMENTAL RESULTS AND ANALYSIS OF SOFTWARE PROJECT COST ESTIMATION USING ANT COLONY SYSTEM

In this paper the proposed of the project data questionnaire we have described the date of the first cost effecting factor. The table 1 shows the Understanding and Consistency of Business Objectives for the Project and Product data. The range of the which the business objectives for the project and product and distinctly understood, and The discerning of objectives between the project team member and the customer and coherent data.

Table 1. The Understanding and Consistency of Business Objectives for the Project and Product

Q	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
1	1	2	2	2	2	3	2	2	2	3	3	3	2	3	3	2	1	3	2	3
2	4	3	2	3	2	2	3	2	3	2	3	2	2	4	3	2	3	4	3	2
3	4	4	4	4	4	4	4	4	4	3	3	3	4	3	4	4	3	2	3	3
4	2	2	1	2	1	2	2	1	2	2	2	2	2	2	2	1	2	3	1	2
5	3	3	2	3	2	3	3	2	3	2	3	2	3	2	2	2	3	3	2	2
L	14	14	11	14	11	14	14	11	14	12	14	12	13	14	14	11	12	15	11	12

The table 2 shows the Key Project Team Capabilities. The knowing of the key people on the project team using lead analyst, and project manager about the application domain for the project, the process and documentation standard and usual practices to be used on the project data, then using development platform and environment , and design by people. In this paper table 3 shows the Customer Participation. The range of the which the customers are efficiently and promptly performing some of the development activities themselves, providing information, and/or reviewing project documents.

Table 2. The Key Project Team Capabilities

Q	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
1	1	2	2	2	2	2	1	1	2	1	2	1	1	2	1	1	1	2	2	1
2	1	2	2	2	2	2	1	1	2	2	2	1	2	1	2	2	1	2	2	1
3	1	2	2	2	2	2	1	1	2	2	2	1	2	1	2	2	1	2	2	1
4	1	1	1	2	2	1	1	1	2	2	1	1	2	2	1	1	1	1	2	1
5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
L	5	8	8	9	9	8	5	5	9	8	8	5	8	7	7	7	5	8	9	5

Table 3. The Customer Participation

Q	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
1	1	2	2	2	2	2	1	1	2	1	2	1	1	2	1	1	1	2	2	1
2	1	2	2	2	2	2	1	1	2	2	2	1	2	1	2	2	1	2	2	1
3	1	2	2	2	2	2	1	1	2	2	2	1	2	1	2	2	1	2	2	1
4	1	1	1	2	2	1	1	1	2	2	1	1	2	2	1	1	1	1	2	1
5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

The table 4 shows the Mixed Teams data and table 5 shows the requirements volatility. The range of the which the agreed upon requirements are required to change over time during the project.

Table 4. The Mixed Teams

Q	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
1	2	2	2	2	2	2	3	3	2	3	2	2	3	3	2	2	2	2	2	3
2	1	2	2	2	1	2	1	2	1	2	2	1	2	2	2	2	2	1	2	2
3	1	1	1	2	2	1	1	1	1	2	2	1	1	1	1	2	1	1	1	2
4	1	2	2	2	2	2	1	1	2	2	1	2	2	2	1	2	2	1	1	2
L	5	7	7	8	7	7	5	7	6	9	7	6	8	8	6	8		5	6	9

Table 5. The Requirements Volatility

Q	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
1	1	1	1	1	1	1	1	1	1	2	2	1	1	2	2	2	1	1	1	1
2	1	3	3	3	3	3	3	3	3	2	3	3	2	2	3	1	3	3	3	1
3	4	4	4	4	4	4	3	4	3	3	4	4	3	3	4	4	4	4	4	4
4	3	3	3	3	3	3	3	3	3	2	3	3	2	2	3	3	3	3	3	3
5	1	2	2	2	2	2	1	1	2	1	1	1	1	2	1	2	1	1	1	1
L	10	13	13	13	13	13	11	12	12	10	13	12	9	11	13	12	12	12	12	10

Table 6. The Development Schedule Constraints

Q	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
1	1	2	2	2	2	2	1	2	1	2	3	2	2	1	2	3	1	2	3	1
L	1	2	2	2	2	2	1	2	1	2	3	2	2	1	2	3	1	2	3	1

In this paper table 6 shows the Development Schedule Constraints of the project. The extent to which a reasonable project schedule is compressed without changing any of the stated requirements. The table 7 shows the Meeting Reliability Requirements of the project. The amount of extra heeds beyond what is stipulated in the organization common practices that is essential to meet the reliability requirements for the part of the system developed by the organization.

Table 7. The Meeting Reliability Requirements

Q	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
1	4	2	2	2	2	2	3	2	3	4	3	2	2	3	3	2	4	4	3	2
L	4	2	2	2	2	2	3	2	3	4	3	2	2	3	3	2	4	4	3	2

The table 8 shows the Meeting Usability Requirements of the project. The amount of extra heeds beyond what is stipulated in the organization common practices that is essential to meet the usability requirements for the part of the system development by organizing.

Table 8. The Meeting Usability Requirements

Q	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
1	1	1	1	2	3	3	2	1	2	1	1	3	3	3	3	2	3	1	3	1
2	4	4	4	3	3	3	3	4	3	4	4	3	3	3	3	3	3	4	3	4
L	5	5	5	5	6	6	5	5	5	5	5	6	6	6	6	5	6	5	6	5

The table 8 shows the meeting performance requirements. The amount of extra heeds beyond what is stipulated in the organization common practices that is essential to meet the performance (i.e., Response time, execution time, and memory usage) requirements for the part of the system development project.

Table 9. The Meeting Performance Requirements

Q	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
1	1	2	2	2	2	2	1	1	2	1	2	1	1	2	1	1	1	2	1	1
L	1	2	2	2	2	2	1	1	2	1	2	1	1	2	1	1	1	2	1	1

Table 10. The Disciplined Requirements Management

Q	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
1	1	2	2	2	2	2	2	2	2	2	1	2	2	2	1	2	1	2	2	1
2	1	1	1	1	2	1	1	2	1	1	1	2	2	2	2	2	1	2	1	2
3	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1
4	1	1	1	1	1	1	2	1	2	1	1	1	2	1	1	1	1	1	2	1
5	3	2	2	2	2	2	2	2	3	2	3	2	2	2	2	3	3	2	2	2
L	7	7	7	7	9	7	8	8	9	7	7	8	9	9	7	9	7	8	8	7

In this paper table 10 shows the Disciplined Requirements Management of the project. The process that is needed for managing transformation in requirements beyond what is considered the organizational common process using project. In this paper proposed technique after so many numbers of Iteration we have found the probability of that particular factor. This process had repeated for the entire factor and we have computed the probability of the each factor. The table 11 shows the representing probabilities of the each factor.

Table 11. The Representing Probabilities of the factor

S. No.	Factors	Probability
1.	Understanding and consistency of Business Objectives	0.7391
2.	Key Project Team Capability	0.80117
3.	Customer Participation	0.9775
4.	Mixed Team	0.94017
5.	Requirement Volatility	0.69112
6.	Development Schedule Constraints	0.84874
7.	Meeting Reliability Requirements	0.99902
8.	Meeting Usability Requirements	0.82753
9.	Meeting Performance Requirements	0.9971
10.	Disciplined requirement management	0.9775

In this paper we are proposing a cost estimation model is combining these probabilities which are performed by Bayesian Cause Effect Model shows figure 6.

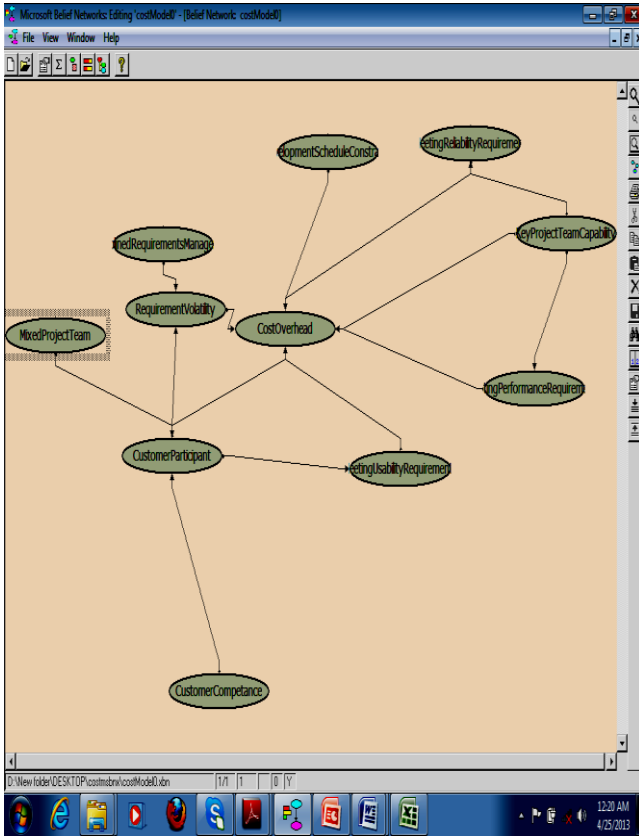


Figure 6. Bayesian Cause –Effect Model

After this step we will get the Cost Overhead. It is expressed as an additional percentage on top of the cost of a project run under optimal conditions. The Cost Overhead is intended to capture the deviation from nominal to actual projects. The next step is to compute the productivity that can be computed by following expression

$$P = \beta_0 - (\beta_1 \times CO)$$

Where CO is the cost overhead and β_0 is the productivity of the nominal project, and β_1 is the ratio of CO and P.

$$\text{Effort} = \alpha \times \text{Size}$$

$$\text{Where } \alpha = \frac{1}{\beta_0 - (\beta_1 \times CO)} \quad \frac{1}{P}$$

Effort in person months can be computed by this expression. The cost overhead is defined as an additional percentage with the cost at the top of the cost of a project run under optimal condition. We can say that overhead is directly proportional to the deviation from nominal cost to the actual cost of the project. To identify the most important cost drivers we choose the experienced project managers, user and analyst related to organization for giving rank to all the cost affecting factors.

The output of this step is a minimal set of cost drivers that have the largest impact on the cost of projects in the local environment. The ranking of project cost drivers all respondents result and the ranking of project cost drivers experienced respondents result are shown in figure 7 and figure 8.

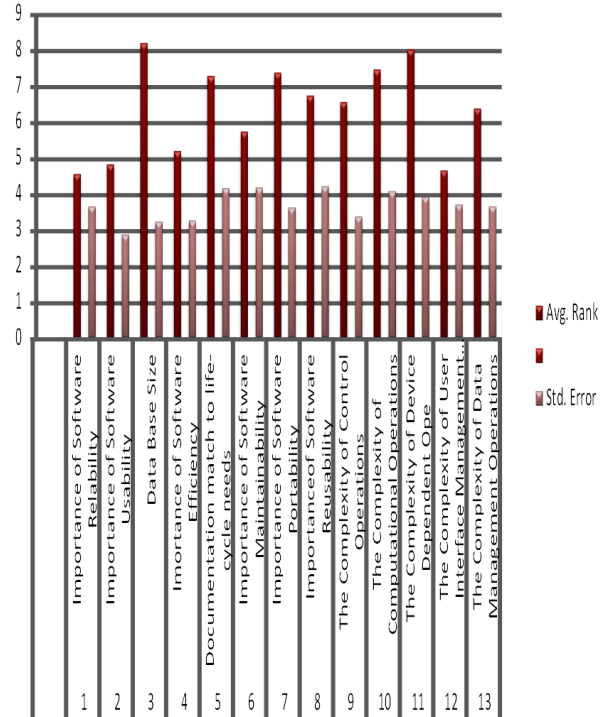


Figure 7. The Ranking of Project Cost Drivers All Respondents Result

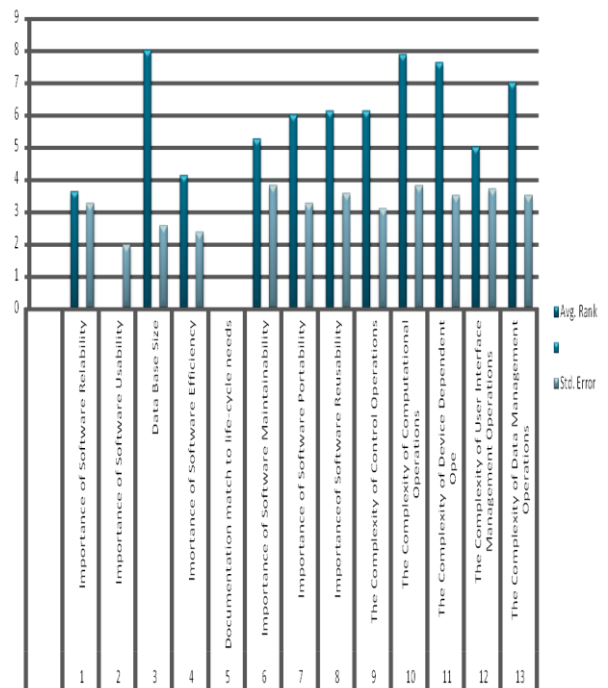


Figure 8. The Ranking of Project Cost Drivers Experienced Respondents Result

7. CONCLUSION

The software cost estimation is the process of predicting the effort required to develop software systems. In this paper, we compute the probability of each selected factors by ant colony system after that combines the value of these factors to calculate the cost overhead for the project by using Bayesian belief network. Our computation gives the Cost Overhead that depends on various factors. Till date Ant Colony Optimization Algorithm has Provided solutions for the problems that have multiple solution and user are interested in best solution. This algorithm provides a proper heuristic for the problem and computes the best possible. The main objective of the in this paper to compute the similarities and differences among various experienced project managers. It gives the solutions in terms of probability, i.e. The most likely occurred solution and the best solution. It was first introduced In Traveling Salesman Problem for finding the minimum cost path. We have mapped our problem in a simple graph by using a questionnaire. That gives the minimum length path, the path that has a minimum deviation from the nominal project for each factor. This can be seen in the validation phase. After this we have computed probability by using MSBNX tool that computes the overall impact of the overhead. That can be further converted in productivity and later on in an effort this paper provides a way to find the best and nearly occurred solution and we have calculated the probability of occurrence of 12 factors on the basis of data collected by project data questionnaire as the amount of data increases the result will be more accurate. The proposed method will work more effectively and efficiently if we will use Ant colony System for combining the probability with Bayesian. We can make the computation task easy and less time consuming by automating it by the tools. Bayesian computation gives better result when we perform it from the Ant Colony System.

8. REFERENCES

- [1] Barry Boehm, Chris Abts, “Software Development Cost Estimation Approaches – A Survey1” , University of Southern California Los Angeles, CA 90089-0781 Sunita Chulani IBM Research 650 Harry Road, San Jose, CA 95120.
- [2] M. Dorigo, V. Maniezzo, and A. Colomi, “ Ant System: Optimization by a colony of cooperating agents” IEEE Transactions on Systems, Man, and Cybernetics – Part B, vol. 26, no. 1, pp. 29–41, 1996.
- [3] K. M. Sim and W. H. Sun, 2003, “Ant colony optimization for routing and load-balancing: Survey and new directions ” IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans.
- [4] V. A. Cicirello and S. F. Smith, 2001, “Ant colony control for autonomous decentralized shop floor routing ” in Proceedings of the International Symposium on Autonomous Decentralized Systems. IEEE Computer Society Press, pp. 383–390.
- [5] Briand, L.C., Wiczorek, I. “Software Resource Estimation” , Marciniak J.J. (ed.), Encyclopedia of Software Engineering, vol. 2, John Wiley & Sons, 2002, pp. 1160-1196.

- [6] Doerner K, Gutjahr WJ, Hartl RF, Strauss C, Stummer C. Pareto, “ ant colony optimization: A metaheuristic approach to multiobjective portfolio selection ” , Ann Oper Res 2004 , vol 131, pp 79–99.
- [7] Dorigo M, Stützle T. Ant Colony optimization. Cambridge, MA: MIT Press; 2004.
- [8] T. St utzle and H.H. Hoos, “Improving the Ant System: A detailed report on the MAX–MIN Ant System” FG Intellektik, FB Informatik, TU Darmstadt, Germany, Tech. Rep. AIDA–96–12, Aug. 1996.
- [9] Steven S. Vicinanza, Tridas Mukhopadhyay, Michael J. Prietula, “Software-Effort Estimation: An Exploratory Study of Expert Performance”, Energy Management Associates, Inc. 100 Northcreek Atlanta, Georgia S0327
- [10] Lionel C. Briand, Khaled El Emam, and Frank Bomarius Fraunhofer, ”COBRA: A Hybrid Method for Software Cost Estimation, Benchmarking, and Risk Assessment”, IESE Sauerwiesen 6D-67661 Kaiserslautern,Germany.
- [11] Jones, Capers, “Applied Software Measurement: Assuring Productivity and Quality”.2ed. McGraw-Hill, 1996.
- [12] Steven S. Vicinanza,Tridas Mukhopadhyay, Michael J. Prietula, “Software-Effort Estimation: An Exploratory Study of Expert Performance”, Energy Management Associates, Inc. 100 Northcreek Atlanta, Georgia S0327.
- [13] Yunsik Ahn, Jungseok Suh, Seungryceol Kim, Hyunsoo Kim, April 2003, “The Software maintenance project effort estimation model based on function points”, Journal of Software Maintenance & Evolution: Research and Practice, vol 15, issue 2.
- [14] Tridas Mukhopadhyay, Steven S. Vicinanza and Michael J. Prietula, “Examining the Feasibility of a Case-Based Reasoning Model for Software Effort Estimation”, MIS Quarterly, Vol. 16, No. 2 June, 1992.
- [15] L.M. Gambardella and M. Dorigo, “Solving symmetric and asymmetric TSPs by ant colonies,” in Proc. IEEE International Conference on Evolutionary Computation (ICEC’96),T. Baeck et al., Eds. IEEE Press, Piscataway, NJ, pp. 622–627, 1996.
- [16] M. Reimann, K. Doerner, and R.F. Hartl, 2004, “D-ants: Savings based ants divide and conquer the vehicle routing problem,” Computers & Operations Research, vol. 31, no. 4, pp. 563–591.

AUTHOR’S PROFILE

Nikhata Akhtar Assistant Professor in the Department of Computer Science & Engineering Integral University, Lucknow, India. She has authored a number of different journal and paper. His research interests include Soft Computing, Artificial Neural Network, Cryptography, Pattern Matching & Recognition, Artificial Intelligence, Network Security, Fuzzy Logic, Network and Database. He is a member of IEEE.