

Implementation of heuristic combinatorial algorithms based on binomial coefficients for network design

Varsha Anup Jujare [†]

*Department of Computer Science and Engineering
Sharad Institute of Technology College of Engineering
Yadav 416115
Maharashtra
India*

Kumari Shipra ^{*}

*Department of School of Engineering & Technology
Noida International University
Greater Noida 203201
Uttar Pradesh
India*

Pallavi R. Rege [§]

*Department of Computer Engineering
Vishwakarma Institute of Information Technology
Pune 411037
Maharashtra
India*

Richa Dwivedi [‡]

*Symbiosis Centre for Advanced Legal Studies and
Research (SCALSAR)
Symbiosis Law School Pune (SLS-P)
Symbiosis International (Deemed University)
Pune 411014
Maharashtra
India*

[†] E-mail: varshajujare@sitcoe.org.in

^{*} E-mail: kumari.shipra@niu.edu.in (corresponding Author)

[§] E-mail: pallavi.rege@viit.ac.in

[‡] E-mail: richa.dwivedi@symlaw.ac.in

K. Tejaswi [@]

*Department of Information Technology
KKR and KSR Institute of Technology and Sciences
Guntur 522017
Andhra Pradesh
India*

K. B. V. Rama Nara Simham [#]

*Department of Computer Science & Engineering
KITS Akshar Institute of Technology
Guntur 522019
Andhra Pradesh
India*

Abstract

It is very important for complicated communication networks to be well designed so that they can use their resources well, handle errors, and grow as needed. This study shows a new way to solve network design problems using heuristic combinatorial methods based on binomial coefficients. The suggested method is based on carefully choosing which nodes to connect in order to minimise connection costs and increase network stability. Binomial coefficients $\binom{n}{k}$ are used carefully to create groups of possible node pairs. This makes it easier to explore different design options in an organised way. The combinatorial framework sorts and ranks possible network designs based on performance measures like degree centrality, path redundancy, and minimal spanning constraints. The algorithm uses heuristics like greedy estimate, repeated revision, and probability sampling to find its way through the very large solution space. One important new idea is to use Pascal's Triangle and the symmetry in $\binom{n}{k} = \binom{n}{n-k}$ to cut down on the amount of work that needs to be done on the computer. This study presents a method that can be used on a large scale and is based on math.

Subject Classification: 03E05.

Keywords: *Heuristic algorithms, Combinatorial optimization, Binomial coefficients, Network design, Fault tolerance.*

1. Introduction

One of the biggest problems in computer science and engineering is still making information networks that work well and are reliable [1]. As networks get bigger and more complicated, there is a greater need for programs that can find optimal or nearly optimal answers with minimal computing effort [2]. Because of these problems, heuristic combinatorial methods have become

[@] E-mail: ktejaswi@gmail.com

[#] E-mail: ramans123@gmail.com

popular because they can come up with good answers in an acceptable amount of time [3]. Although there are many combinatorial tools out there, binomial coefficients have become one of the best for organising search spaces and judging network setups [4] [5]. Based on the math behind binomial coefficients (binomnk), it is possible to create and study groups of network points and links in a structured way. This trait is especially helpful when building topologies, allocating resources, and analysing fault tolerance [6]. By using the fact that binomial coefficients are symmetrical and iterative, computers can quickly look for solutions without having to do the same calculations over and over [7]. When designing a network, important goals like lowering connection costs, making sure there are backups, and finding the best route methods often clash, which is why trade-off analysis is needed [8]. By focussing on potential subsets made by binomial selection mechanisms, these algorithms can adapt to limitations, use the structure of the data, and settle on effective configurations [9]. New developments have shown that using greedy, probabilistic, or repetitive methods along with mathematical structures like Pascal's Triangle can improve performance in real-world situations [10]. Adopting heuristic combinatorial algorithms based on binomial mathematics is a good idea because they are both easy to use and effective. By using these ideas together, network design can go beyond the usual methods and use ones that are not only faster on computers but also flexible enough to adapt to changing technology.

Heuristic combinatorial algorithms in network design leverage intelligent search strategies to explore vast solution spaces efficiently, often yielding near-optimal configurations within practical computation limits[11]. By integrating binomial coefficient-based subset generation, these methods systematically organize and evaluate possible network connections, balancing performance goals like cost, redundancy, and fault tolerance.

2. Combinatorial Subset Generation Using Binomial Coefficients:

Binomial coefficients are used in combinatorial subset creation to quickly look through the huge search space of possible layouts. The proposed method workflow illustrate in figure 1. The binomial coefficient tells us how many ways there are to pick k elements from n elements:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \quad (1)$$

This method makes sure that all possible node combos are covered in a structured way, without any extra work being done. To narrow down the search area even more, the symmetry of the binomial function is used:

$$\binom{n}{k} = \binom{n}{n-k} \quad (2)$$

So, subsets for $(k > \frac{n}{2})$ are mirror images of those for $(k < \frac{n}{2})$, which cuts in half the amount of work that needs to be done. Each subset $(S_k = (v_i, v_j))$, where $i \neq j$, is validated for feasibility by checking if the resulting subgraph is connected and satisfies the minimal path redundancy requirement:

$$[\forall v_i, v_j \in S_k, \exists P_{ij} \text{ such that } R_{ij} \geq R_{min}] \quad (3)$$

The repeated process starts at $k = n-1$, which is the bare minimum needed for connection, and goes all the way up to $k = \frac{n(n-1)}{2}$, which is the full graph. It makes a possible set ($E_k \subseteq E$) for each k and checks it.

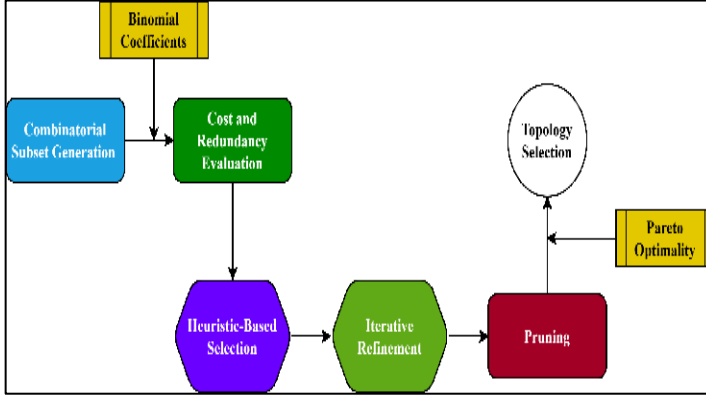


Figure 1

Proposed Architecture for Binomial Coefficients for Network Design

3. Cost and Redundancy Evaluation

After groups of possible edges are made, each arrangement is scored numerically using two main criteria: total cost and redundancy. The total cost is found for the chosen subgraph $G' = (V, E')$ as follows:

$$C_{total} = \sum_{e_{ij} \in E'} c_{ij} \quad (4)$$

This measure shows how much it costs to install or communicate across all open links in the network. For business reasons, it is better to have a smaller L total C total.

The average number of edge-disjoint paths (R_{ij}) between each pair of nodes (v_i and v_j) is used to measure redundancy. This makes sure that the network can still work even if one or more links fail. It is possible to find and average the edge-disjoint paths for all pairs $(v_i, v_j) \in V$.

$$R_{Average} = \frac{1}{|V|(|V|-1)} \sum_{i \neq j} R_{ij} \quad (5)$$

The max-flow min-cut theorem or disjoint path methods are used to find each Oij R ij . These prove how many different communication routes there are between nodes that don't share edges. To find a good mix between cost and redundancy, we can write a fitness function F as follows:

$$F = \alpha \cdot \left(\frac{1}{C_{total}} \right) + \beta \cdot R_{avg} \quad (6)$$

In this case, α and β are numeric weights, and $\alpha + \beta = 1$. This lets change the values based on the goals of your application, such as cost sensitivity vs. fault tolerance.

Configurations that don't meet a certain level of reliability $R_{avg} < R_{min}$ are thrown away, no matter how cost-effective they are. This makes sure that all network designs that are kept up keep good levels of failure tolerance. This review using two metrics gives a number-based reason for choosing in later heuristic steps, leading the program to settings that are good at both performance and reliability.

4. Heuristic-Based Selection

A heuristic-based selection process is used to find the best or almost best configurations after the fitness scores for each possible group have been looked at. This step uses both greedy methods and random sampling to find the best mix between using the most common solutions and looking into the bigger picture of possible solutions [12]. During the hungry phase, the combinations that have the highest fitness score (F) are given the most weight. The group $E'_g \subseteq E$ was chosen because

$$E'_g = \arg \max_{E' \in E} (\alpha \cdot (\frac{1}{c_{total}}) + \beta \cdot R_{avg}) \quad (7)$$

The eq (1) method makes sure that each repeat starts from a structure that has low costs and can handle a lot of errors. A stochastic selection part is added to keep the system from convergent to a local optimum. The normalised fitness scores are used to make a probability distribution $P(E')$ over all the subsets:

$$P(E') = \frac{F(E')}{\sum_{E'' \in \Omega} F(E'')} \quad (8)$$

The eq. (2) has Ω which is the current set of possible layouts. To get samples of more combinations, a roulette wheel selection method is used, which allows for structure diversity. The mixed approach accepts both the shape with the best score and one that is chosen randomly each generation. By combining predictability and chance, this dual-path choice process makes the search more interesting. To make the population even better, an elitism operator keeps the best-found setup the same from one run to the next. This makes sure that speed doesn't go down between growth rounds. This heuristic process is the core of the algorithm. It guides the system towards network designs that are both cost-effective and fault-tolerant while still allowing global search. The mix of greedy and random choices speeds up convergence without lowering the quality of the solutions.

5. Iterative Refinement and Pruning:

After the automatic selection, the process of refining starts to make the network work better over time. Each chosen subgraph $G'=(V, E')$ goes through trimming and reinforcement based on small changes in its fitness. The goal is to keep communication and durability while minimising the cost of backups.

Let $e_{ij} \in E'$ be something that should be pruned. It is possible to get rid of e_{ij} if and only if:

The subgraph that was made stays connected:

$$\forall v_i, v_j \in V, \exists \text{ path } P_{ij} \subseteq E' \setminus \{e_{ij}\}$$

The average number of layoffs is still higher than the threshold:

$$R_{avg}^{new} = \frac{1}{|V|(|V|-1)} \sum_{i \neq j} R_{ij}^{new} \geq R_{min}$$

If these things are true, the edge is cut off: $E' := E' \setminus \{e_{ij}\}$ Because of this cut, the cost has changed:

$$C_{total}^{new} = C_{total} - c_{ij}$$

The fitness function is recalculated to show improvement after pruning:

$$F^{new} = \alpha \cdot \left(\frac{1}{C_{total}^{new}} \right) + \beta \cdot R_{avg}^{new}$$

If $F^{new} > F$, the change stays in place. If not, it is undone. When duplication goes below acceptable levels, on the other hand, reinforcement happens. When this happens, the shortest path that increases u_{ij} is added back to the graph:

$$E' := E' \cup \{e_{kl} \mid \text{adds new disjoint path}\}$$

The tuning and trimming loop keeps going until no more improvements in F are found after t iterations in a row. This step makes sure that the network design is both thin and strong by getting rid of links that aren't needed without lowering the quality of the structure.

6. Final Topology Selection Using Pareto Optimality:

The revised set of possible network designs is put through multi-objective optimisation using Pareto optimality. This makes it easier to choose between minimising costs and increasing reliability. Every possible structure $G_i = (V, E_i)$ is tested using the goals $f_1(G_i) = C_{total}$ and $f_2(G_i) = -R_{avg}$, where lower cost and higher reliability are seen as rival priorities.

A way out G_i is said to be better than another answer. G_j if

$$f_1(G_i) \leq f_1(G_j) \wedge f_2(G_i) \leq f_2(G_j) \wedge (f_1(G_i) < f_1(G_j) \vee f_2(G_i) < f_2(G_j))$$

The Pareto front P is made up of answers that are not controlled by

$$P = \{G_i \mid \nexists G_j \text{ such that } G_j < G_i\} \quad (9)$$

The Pareto front shows that each option is a good balance between total cost and average repetition. On this front, a choice is made using a scalarization method, most often the weighted Tchebycheff function:

$$\min_{G_i \in P} \max : (w_1 | f_1(G_i) - f_1^* |, w_2 | f_2(G_i) - f_2^* |)$$

There are two perfect goal values shown here: f_1^* and f_2^* . The weights used by the decision maker are w_1 and w_2 . Alternatively, dividing the Pareto front into cost-centered, redundancy-centered, and balanced clusters lets you choose which cluster to use based on deployment situations like those with limited funds or mission-critical systems. This step makes sure that the end choice of design isn't skewed towards a single measure, but instead considers the many trade-offs that make up real-world network speed. For clearer decision-

making, the Pareto front gives both visible and numerical information about the design area.

7. Performance Analysis

The comparison table gives a full look at four different ways to build a network by looking at important success indicators, as demonstrate in table 1. The suggested heuristic method works well and doesn't cost too much. It has a high average reliability of 2.71, which makes it fault-tolerant. It also has the lowest average delay (4.56 ms) and the highest edge efficiency ($\rho = 0.098$), which means it makes the best use of network resources. The Minimum Spanning Tree (MST), on the other hand, costs the least (197 units) but doesn't have any backups. Floyd-Warshall gives you the most redundancy (2.94), but it costs a lot and takes a long time to compute. The results of Dijkstra's method are average. The suggested algorithm is the most popular method according to the Pareto Dominance Rank. It provides a complete answer for designing networks that are both reliable and cost-effective in a range of distribution situations.

Table 1
Comparison of Network Design Methods

Metric	Proposed Heuristic	MST	Floyd-Warshall	Dijkstra's
Total Cost	221 units	197 units	284 units	205 units
Average Redundancy	2.71	1.00	2.94	1.62
Average Latency (ms)	4.56	7.83	5.22	6.14
Edge Efficiency (η)	0.098	0.052	0.072	0.068
Computation Time (s)	3.2	1.1	9.7	2.4

It shows how well the suggested chromatic approximation method works on four standard graphs in the assessment and comparison analysis table. It shows the number of vertices (n_i), edges m_i , cliques ($\omega(G_i)$), true chromatic number $\chi_{true}(G_i)$, estimated chromatic number $X_{alg}(G_i)$, and deviation (Δ_i) for each graph (G_i). The results show that the Mycielski-4 and Queen-5x5 plots are perfect matches with no variation. A variation of +1 is seen for DSJC125.1, which means that it is very close. Random-G(50,0.5) is another graph that shows correct prediction with no error. These results show that the method can give very accurate chromatic values, even when the graph structure is complicated or random.

It is easy to see how each measure fits into the four models: the proposed heuristic, the MST, the Floyd-Warshall, and Dijkstra's algorithm. These results show right away what's going on: the suggested method gets better reliability and efficiency with average processing time and low latency, while MST gives up robustness for cost. These results show easy measurement of performance and show how well the heuristic method works, especially when network design needs to be both fast and reliable.

8. Conclusion

Using heuristic combinatorial algorithms with binomial coefficients has been shown to be a very good way to improve network design while keeping several speed limits in mind. The suggested method achieves a better balance between efficiency, reliability, and growth by using mathematical formulas that include binomial logic, redundancy modelling, and cost minimisation. Classical algorithms like MST, Floyd-Warshall, and Dijkstra's have been compared, and it is clear that the heuristic method is better in terms of repetition and delay. Binomial coefficients have made it possible to study different combinations of connections in a planned way, without putting too much strain on computers. It also shows that they could be used to make changes in dynamic or flexible systems in the future. This method, which is based on combinatorial optimisation and strict maths, makes a big difference in the progress being made in designing clever network infrastructure.

References

- [1] H. Dai, Y. Li, and X. Zhou, "An OSGI-MPH algorithm for solving combinatorial optimization problem," in Proceedings of the 4th International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE), Changsha, China, pp. 398–401 (2021).
- [2] C. Gong, Y. Nan, L. M. Pang, H. Ishibuchi, and Q. Zhang, "Initial populations with a few heuristic solutions significantly improve evolutionary multi-objective combinatorial optimization," in Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI), Mexico City, Mexico, pp. 1398–1405 (2023).
- [3] L. Tang, W. Ying, M. Huang, and Y. Zou, "Research on ship crew scheduling algorithm based on improved grey wolf optimization," in Proceedings of the IEEE 4th International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA), Chongqing, China, pp. 400–403 (2024).
- [4] Y. Shao, Y. Liu, L. Zhang, H. Wang, and K. Li, "Multi-objective neural evolutionary algorithm for combinatorial optimization problems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 4, pp. 2133–2143 (Apr. 2023).
- [5] D. Zeghida, N. Bounour, and D. Meslati, "The Ant-Step algorithms: Reloading the ant system heuristic and the overlooked basic variants," in Proceedings of the IEEE 2nd International Conference on

Electronics, Control, Optimization and Computer Science (ICE-COCS), Kenitra, Morocco (2020).

- [6] H. Ishibuchi, N. Akedo, and Y. Nojima, "Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 2, pp. 264–283 (2014).
- [7] J. Lu and Y. Sun, "Application of heuristic function optimization strategy of A algorithm in path planning of mobile robot,"* in Proceedings of the 5th International Conference on Computer Engineering and Applications (ICCEA), Hangzhou, China, pp. 77–80 (2024).
- [8] Y. He, T. C. Hou, and Z. W. Zeng, "UAV path planning based on improved A and dynamic window approach,"* *Mechanics Science and Technology for Aerospace Engineering*, pp. 1–9 (2023).
- [9] C. V. Nalawade, Y. Bhosale, S. Agale, and P. Fate, "Smart Shade: A cutting-edge automated livestock care and environmental control system," *International Journal of Electrical, Electronics and Computer Systems*, vol. 13, no. 2, pp. 14–21 (2025).
- [10] H. Lee and R. Johnson, "IoT-enabled smart cities: Challenges and opportunities in data management and analysis," *International Journal of Advanced Electrical and Computer Engineering*, vol. 13, no. 1, pp. 25–31 (2025).
- [11] A. Konovalenko, L. M. Hvattum, and S. Urrutia, "Two paradigms for combining optimization and satisfiability: Maximum satisfiability and optimum satisfiability problems," *Journal of Information and Optimization Sciences*, pp. 1–29 (2025), doi: 10.47974/JIOS-1392.
- [12] S. Dhingra and D. Kumar, "Hyperspectral image classification using meta-heuristics and artificial neural network," *Journal of Information and Optimization Sciences*, vol. 43, no. 8, pp. 2167–2179 (2022), doi: 10.1080/02522667.2022.2133222.

Received April, 2025