Beyond the Dipper: Algorithmic Constellation Crafting via Branch and Bound

Najwa Kahani Fatima - 13523043[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*najwanewa1010@gmail.com, 13523043@std.stei.itb.ac.id*

***Abstract*—This research employs a branch and bound algorithm to systematically generate star groupings that align with perceptual and geometric principles associated with constellation recognition. The algorithm evaluates subsets of stars based on five key visual heuristics. First, magnitude influences selection priority—brighter stars (lower magnitude values) are favored as anchor points due to their visual prominence. Proximity is measured as the angular distance between stars, with closer groupings considered more perceptually coherent. Convexity is assessed by the internal angles formed by star triplets, with smoother, near-180° configurations prioritized for visual harmony. To ensure perceptual balance, the algorithm also promotes even spacing, penalizing star chains with large variance in angular separation. Finally, good continuation is modeled by rewarding paths with consistent directional flow and minimal angular deviation. The branch and bound approach enables efficient pruning of suboptimal star combinations early in the search process, significantly reducing computational complexity while maintaining alignment with visual grouping principles. This method not only optimizes for visual clarity but also mimics human cognitive biases in constellation perception.**

***Keywords*— branch and bound, constellation, star**

I. Introduction

In our daily lives, we are often confronted with an abundance of choices. Beneath each decision lies an underlying effort to identify the most favorable outcome, which is a process known as optimization. This principle is central to numerous computational algorithms, enabling solutions to complex problems across various domains. One particularly powerful technique in this context is the Branch and Bound algorithm, which systematically explores the solution space by breaking a large problem into smaller sub-problems and eliminating those that cannot yield better solutions than the current best.

To illustrate the potential of this method, imagine looking up at the night sky, a vast expanse dotted with countless stars. From this seemingly infinite canvas, how might one design a new constellation that is not only aesthetically pleasing but also adheres to specific criteria, such as visual recognizability or scientific relevance? This paper explores the use of Branch and Bound to identify an optimal subset of stars from a large dataset, aiming to construct a constellation that maximizes a set of defined metrics while satisfying constraints such as a limit on the number of stars or the total angular area covered.

Through this lens, this paper examines a classical optimization technique can be applied to a novel and imaginative problem, bridging computation and celestial design in a unique and meaningful way.

II. Branch and Bound

Branch and Bound (B&B) is a widely used algorithmic framework for solving combinatorial and discrete optimization problems where the solution space is too large to explore exhaustively. The method is particularly effective for problems such as integer programming, the traveling salesman problem (TSP), and constrained subset selection, where optimality is required but brute-force approaches are computationally infeasible. The B&B approach systematically explores the solution space by dividing it into smaller subproblems (branching) and eliminating regions that cannot contain the optimal solution based on bound estimates (bounding) [1].

The search process in B&B is typically visualized as a tree, where each node represents a partial or complete solution. The method ensures correctness and optimality by expanding only promising branches while pruning others early, thereby dramatically reducing the number of nodes that must be evaluated. This selective exploration makes the algorithm more efficient than exhaustive search, especially when paired with effective bounding techniques.

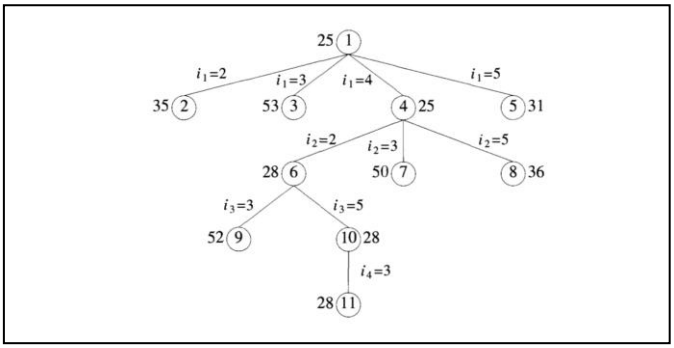


Figure 1. Example of Branch and Bound Tree on Traveling Salesman Problem [2]

A. Least Cost Search

Least Cost (LC) Search is a node-selection strategy often used within the Branch and Bound framework. In LC search, nodes in the search tree are explored in the order of their estimated cost, specifically, the node with the lowest cost is expanded first. This cost is determined using a bounding function, which must be admissible (i.e., never overestimating the true cost to a solution) to guarantee optimality [3].

The general steps for LC Branch and Bound are as follows:

1. Initialization: Place the root node representing the initial solution in a priority queue.
2. Node Expansion: At each iteration, remove the node with the lowest cost estimate from the queue.
3. Bounding: If the cost of the node exceeds the current best-known solution, discard it.
4. Branching: Generate child nodes by extending the current partial solution.
5. Solution Update: If a complete solution is found with a lower cost, update the best-known solution.
6. Repeat until the queue is empty or the optimal solution is confirmed.

This strategy is efficient for problems where the bounding function closely approximates the real cost, enabling aggressive pruning and rapid convergence to optimality [4].

B. Branching and Bounding

The two fundamental components of the B&B algorithm are branching and bounding. Branching refers to the recursive division of the original problem into smaller subproblems by making a decision at each node, such as including or excluding an element. This step expands the search tree and systematically explores the solution space.

Bounding involves estimating the minimum (or maximum, depending on the problem) cost that can be achieved from a given node. If this bound is worse than the cost of an existing complete solution, the node and all of its descendants can be discarded—a process known as pruning. Formally, if the bound $f(n)$ of a node n satisfies $f(n) \geq f^*$, where f^* is the cost of the best known feasible solution, then node n is pruned [5].

In the context of optimization and search algorithms, the cost function $c(i)$ is a key component used to evaluate and prioritize nodes (or partial solutions) during exploration. The formula:

$$c(i) = f(i) + g(i) \quad (1)$$

decomposes the total estimated cost $c(i)$ of a solution path passing through node iii into two parts: 1) $f(i)$ which is the actual cost incurred from the root node to the current node i and this represents the known or accumulated cost of reaching the current state in the search tree, 2) $g(i)$ which is the estimated cost to complete the remaining portion of the solution from node i to a terminal (goal) state [2]. This component is often derived from a heuristic or bounding function that provides a lower-bound estimate of the cost needed to finish the partial solution.

Together, $c(i)$ represents the estimated total cost of a solution path that passes through node i. In a Branch and Bound algorithm using Least Cost (LC) Search, nodes are typically selected for expansion in order of increasing $c(i)$ prioritizing those with the smallest expected total cost. This decomposition is especially important in admissible heuristics where $g(i)$ must never overestimate the true remaining cost in order to guarantee that the first complete solution found is optimal.

In Branch and Bound, $f(i)$ is often the exact cost of partial solutions (e.g., the sum of selected values, distances, or weights), and $g(i)$ may come from a problem-specific relaxation or approximation that underestimates the remaining cost. By combining these, the algorithm can balance exploration and exploitation effectively.

The effectiveness of the Branch and Bound algorithm hinges on the tightness of the bounding function and the branching strategy. A well-designed bounding function significantly reduces the size of the search tree, making the approach scalable for larger problems. Because of its general applicability and theoretical guarantees, Branch and Bound continues to serve as a foundational tool in optimization research and practice [6].

III. The Stars and Constellations

Stars have captivated human curiosity for millennia, serving not only as sources of light in the night sky but also as fundamental objects of scientific study. Each star can be described by its celestial coordinates which define its position on the celestial sphere, along with its properties. Over time, cultures around the world have grouped stars into constellations, recognizable patterns that often reflect mythological figures, animals, or objects. Beyond their cultural significance, constellations also serve practical purposes in navigation, astronomy, and astrophysics, offering a structured way to segment and reference regions of the sky. In modern times, computational methods enable the design and analysis of new constellations based on scientific criteria, such as geometric coherence, visibility, and spatial distribution.

A. The Star Characteristics

In astronomy, stars are luminous celestial objects that populate the night sky. For computational and observational purposes, each star is typically characterized by a set of standard properties. The most common of these include Right Ascension (RA), Declination (Dec), and Apparent Magnitude (Mag) [7][8].

Right Ascension (RA) is analogous to longitude on Earth and measures a star's position along the celestial equator. It is usually expressed in hours, minutes, and seconds, with 24 hours corresponding to a full 360° rotation. Declination (Dec) corresponds to latitude, measuring a star's angular distance north or south of the celestial equator, expressed in degrees. Apparent Magnitude represents the brightness of a star as seen from Earth. A lower magnitude indicates a brighter star. This value is essential in visual filtering, as more prominent stars are often more noticeable and thus preferable for constellation design.

B. Distances Between Stars

To evaluate the spatial separation between two stars on the celestial sphere, the concept of angular distance is essential. It measures how far apart two stars appear from an observer's point of view on Earth, which is critical for constellation design. One effective and numerically stable method to compute small range of distances is the Haversine formula, widely used in spherical geometry.

Given two stars with coordinates (RA_1, Dec_1) and (RA_2, Dec_2), the Haversine formula calculates the angular distance d between them as follows [9]:

$$\Delta RA = RA_2 - RA_1 \quad (1)$$

$$\Delta Dec = Dec_2 - Dec_1 \quad (2)$$

$$\alpha = sin^2\left(\frac{\Delta Dec}{2}\right) + cos(Dec_1) \cdot cos(Dec_2) \cdot sin^2\left(\frac{\Delta RA}{2}\right) \quad (3)$$

$$d = 2 \cdot tan^{-1}2(\sqrt{a}, \sqrt{1-a}) \quad (4)$$

Here, RA and Dec must be expressed in radians. The result d is also in radians and represents the central angle between the two stars as measured from the center of the celestial sphere and serves as a fundamental metric for determining the proximity of stars.

This method offers superior accuracy for small angular distances compared to the spherical law of cosines, making it especially useful in astronomical datasets where stars may be closely spaced. In a practical implementation, this can be written as a function that takes the coordinates of two stars and returns their angular separation. The approach is not only computationally efficient but also avoids rounding errors common in other trigonometric formulas when dealing with very small angles.

In addition to pairwise distance, it is often necessary to evaluate the angle formed between three stars, particularly when analyzing the geometric structure of a constellation. Given three stars A, B, and C, where B is the vertex, and a, b, and c are the angular distances between the corresponding star pairs (with a being the side opposite to vertex B), the angle θ at vertex B can be computed using the Law of Cosines for spherical triangles, as follows [10]:

$$\theta = \arccos\left(\frac{b^2 + c^2 - a^2}{2bc}\right) \quad (5)$$

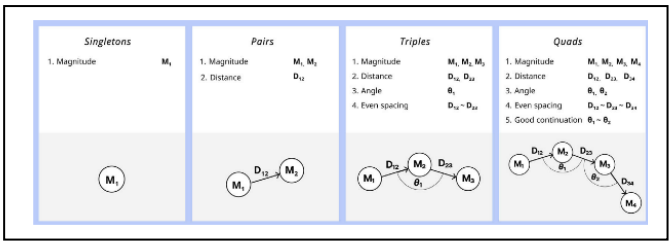
C. The Constellations

The formation and recognition of constellations are strongly influenced by fundamental Gestalt visual principles, which describe how the human perceptual system organizes and interprets visual stimuli into structured patterns. In particular, the principles of proximity, collinearity (or convexity), and good continuation have been shown to guide how individual stars are mentally grouped into coherent constellations. Building on these perceptual foundations, five specific visual and geometric factors have been identified as influential in the crafting and selection of star constellations: magnitude, proximity, convexity, even spacing, and good continuation.

1. Magnitude, defined as the apparent brightness of a star, plays a prominent role, brighter stars (lower magnitude values) are more likely to attract attention and be included in constellations.
2. Proximity, which refers to the angular distance between stars on the celestial sphere, supports the Gestalt principle that spatially closer elements are more likely to be perceived as a group.
3. Convexity is characterized by the internal angle formed between three connected stars, where larger angles approaching 180^o are preferred, producing visually smooth and geometrically stable patterns.
4. Even spacing among stars, i.e., maintaining consistent angular distances within a group, contributes to perceptual balance and facilitates faster recognition of the constellation as a single object.
5. Good continuation reflects the human preference for smooth, consistent directionality in visual paths; constellations where star connections preserve similar angles or follow linear trajectories are more likely to be perceived as cohesive structures.

Interestingly, the influence of these visual principles is not uniform across all constellation sizes. Studies have shown that as the number of stars in a group increases, the influence of magnitude diminishes, especially in larger configurations such as four-star groups (quads), where geometric properties like angle and spacing take precedence in guiding star selection. This shift underscores the dominance of spatial structure over brightness in complex pattern perception, reinforcing the need for multi-criteria optimization in computational constellation design.

Figure 2. The visual principles of each constellation group size [7]



III. PROPOSED METHOD

This paper proposes a method for crafting optimal constellations from a given dataset of stars by implementing five key visual principles derived from perceptual psychology: magnitude, proximity, convexity, even spacing, and good continuation. These principles are quantitatively modeled and integrated into a scoring function that evaluates the visual and geometric quality of a candidate constellation. To efficiently search for the optimal subset of stars under given constraints, the Branch and Bound (B&B) algorithm is employed as the core optimization framework.

The structure of the problem closely resembles the classical Travelling Salesman Problem (TSP) in that it explores combinations of nodes (stars) and evaluates them based on a cost (or score) function. As in the TSP's B&B approach, the method considers all possible paths by branching on star inclusion or exclusion and uses bounding to prune suboptimal branches early, thus avoiding exhaustive enumeration.

To construct visually plausible constellations from a given set of stars, this paper utilizes a Branch and Bound (B&B) approach to search the combinatorial space efficiently. The core idea is to evaluate subsets of stars based on visual grouping principles, while pruning the search tree using upper-bound

estimations to avoid unnecessary computation. The selection process is framed as an optimization problem, where the goal is to find a subset of stars that maximizes a perceptual score under two constraints: a maximum number of stars $N_{\max}$) and a maximum bounding area ($area_{max}$).

A. Visual Principle-Based Scoring

Each candidate constellation subset is evaluated using a multi-objective score function based on five perceptual visual principles with score as follows:

$$score = w_1 M_{avg} + w_2 D_{avg} + w_3 C + w_4 E + w_5 G$$

where M_{avg} is average (negated) magnitude of stars, D_{avg} is average (negated) pairwise angular distance, C is convexity score, E is even spacing score, G is good continuation score, and w_1 to w_5 are weights set by the user for each visual factor. Below N shows number of stars, θ value from Formula (5), d is angular distance from Formula (4), and M is apparent magnitude.

Parameter	**Formula**
Average magnitude (M_{avg})	$M_{avg} = \frac{1}{N}\sum_{i=1}^{N} M_i$
Average angular distance (D_{avg})	$D_{avg} = \frac{1}{N}\sum_{i=1}^{N} d_i$
Convexity value (C)	$C = \frac{1}{N}\sum_{i=1}^{N} 1 - \left\lvert\frac{\pi - \theta_i}{\pi}\right\rvert$
Even spacing value (E)	$Var(d) = \frac{1}{N}\sum_{i=1}^{N}(d_i - \mu)^2$ $E = 1 - \min\left(1, \frac{Var(d)}{V_{max}}\right)$
Good continuation value (G)	$G = \frac{1}{N}\sum_{i=1}^{N} 1 - \left\lvert\frac{\theta_1 - \theta_2}{\pi}\right\rvert$
Bounding Area (A)	$A = (\max RA_i - \min RA_i) \cdot$ $(\max Dec_i - \min Dec_i)$

B. Branch and Bound Strategy

The core search algorithm applies Branch and Bound to explore combinations of stars. Branching involves choosing whether to include or exclude the next star in the current subset. Bounding is used to estimate the upper bound of the score if we continue adding stars to the current partial solution. If this bound is lower than the best-known solution, the branch is pruned.

The decision at each node in the B&B tree is based on formulation (1) where $f(i)$ is the the actual score of the current subset (based on visual principles) and $g(i)$ is the the heuristic estimate (upper bound) of the maximum additional score possible by adding more stars.

The algorithm uses Best-First Search (BFS) Branch and Bound which uses a priority queue to always expand the node with the highest upper bound next. This improves convergence speed, especially in large star sets.

The algorithm prunes paths that violate maximum number of stars N_{max} and bounding area constraint $area_{max}$, or if their potential upper bound is lower than the best score found so far.

IV. PROGRAM EXPERIMENT

This program experiment utilized the Hipparcos Star Catalog as the primary dataset, which contains high-precision data on stellar positions, magnitudes, and identifiers [11]. To validate and visualize specific stars and constellations, the Stellarium desktop application is used as a reference tool for real-world star positioning and appearance. The experiment was implemented in Python, leveraging libraries such as Matplotlib for plotting constellations, and built-in modules like math, heapq, and itertools for performing coordinate transformations, optimizing star selection, and simulating constellation connections. The process involved filtering and transforming the dataset based on visual magnitude and spatial position, mapping these to a 2D coordinate plane, and then using algorithmic logic to construct simplified visual representations of constellations for comparative analysis with Stellarium outputs.

The github repository is attached in the attachment section. Below is the pseudocode of the program:

```
function branch and bound():
 sort all_stars by magnitude (brightest first)
 initialize max-heap with root node:
   subset = []
   score = 0
   area = 0
   upper_bound = estimate upper bound from all_stars
   push (-upper_bound, index=0, subset, score, area,
        path=[]) to heap
   while heap is not empty:
      pop node with highest upper bound
      if subset violates constraints (too many stars
or area too large):
           prune and continue
       if score > best_value:
           update best_value,
best_constellation_subset, and best_path
       if subset is full (reached max stars):
           mark as leaf and continue
       for each star from current index to end:
           new_subset = subset + star
           new_area = bounding box area of new_subset
           if new_area exceeds limit:
               skip star
           new_score = calculate score of new_subset
           remaining_stars = all_stars after current
star
           upper_bound = estimate upper bound from
new_subset and remaining_stars
           if upper_bound <= best_value:
               prune and continue
           push (-upper_bound, new_index, new_subset,
new_score, new_area, updated_path) to heap
    return best_constellation_subset, best_value,
best_path
```

In this part of the experiment, a specific subset of stars from the Hipparcos catalog was selected, identified by the following:

ID	HIP	VMag	RAdeg	DEdeg
0	71974	5.7	220.81	-25.00
1	72197	5.15	221.50	-25.44
2	72323	5.61	221.84	-25.62
3	72357	5.23	221.94	-26.09
4	72378	5.23	221.99	-26.65
5	72420	7.03	222.12	-25.49
6	72488	5.68	222.33	-24.25
7	72979	9.57	223.71	36.40
8	73150	6.91	224.24	-26.28
9	73189	6.94	224.38	-25.44

These stars are chosen from the Stellarium application to ensure positional accuracy and visual consistency. The dataset was filtered to include only these stars, enabling focused analysis on their spatial distribution and relationships. The goal was to reconstruct the constellation formed by these stars and validate its structure against the Stellarium reference. Below is the view of the group of stars above

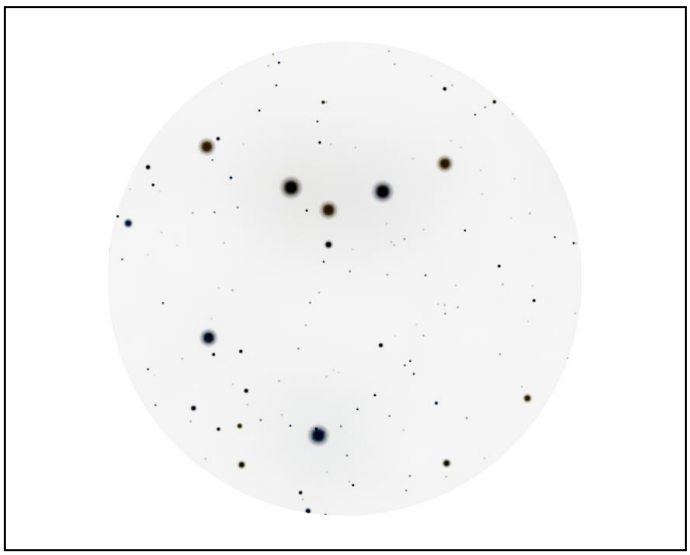


Figure 3. The Group of Star View.

These are the processes to identify the optimal subset of stars forming the most promising candidate constellation:

1. Initialization

The weight for each parameter is set equally to 0.2 with $N_{max} = 8$ and $area_{max} = 50.0$. The algorithm starts with an empty subset (Node 0), corresponding to no selected stars. The initial score is 0.00, with an area of 0.00 and an upper bound (UB) estimate of -0.71. From this node, the algorithm expands by individually adding each of the 10 stars as initial candidates, evaluating their partial scores and corresponding upper bounds. Each resulting node is enqueued for further exploration if its upper bound does not already fall below the current best solution.

2. Node Expansion and Evaluation

Nodes are expanded in order of best estimated upper bound (i.e., "best-first"). At each node, the algorithm:

a. Evaluates the current subset's score (a metric reflecting geometric or photometric criteria),

b. Computes the area covered by the subset,

c. Estimates the upper bound (UB) of the best possible score attainable by any superset of the current subset.

If the node's upper bound is worse than the best score found so far, the node is pruned. Otherwise, the algorithm generates new nodes by attempting to include additional stars, skipping any that would violate area constraints.

3. Star Rejection and Pruning

During traversal, many candidate nodes were pruned:

a. Upper Bound Pruning: Nodes where the upper bound was worse than the best known score (Best = -0.60) were eliminated from consideration. This aggressive pruning significantly reduces the search space.

b. Area Constraint Enforcement: Star 7 was consistently skipped due to exceeding the maximum allowed constellation area. This ensured spatial compactness in the candidate constellation.

4. Constellation Improvement and Optimality

A notable improvement occurred at Node 7 with the subset [1, 3, 2, 6]:

a. Score improved from an initial best of -1.03 (Node 1) to -0.60.

b. This subset achieved the best trade-off between magnitude distribution, spatial cohesion, and angular configuration (as encoded in the scoring function).

Once this optimal subset was found, subsequent branches whose UB could not surpass -0.60 were pruned, solidifying the result's optimality. Below is the final result of the constellation construction with total score -0.6 and total area 1.52.

ID	RA	Dec	Mag
1	221.50	-25.44	5.15
3	221.93	-26.08	5.23
2	221.84	-25.62	5.61
6	222.32	-24.25	5.68

These are the step-by-step illustration for every time a star is chosen into solution set:

Figure 4. Step 2 and 3 of the Construction.

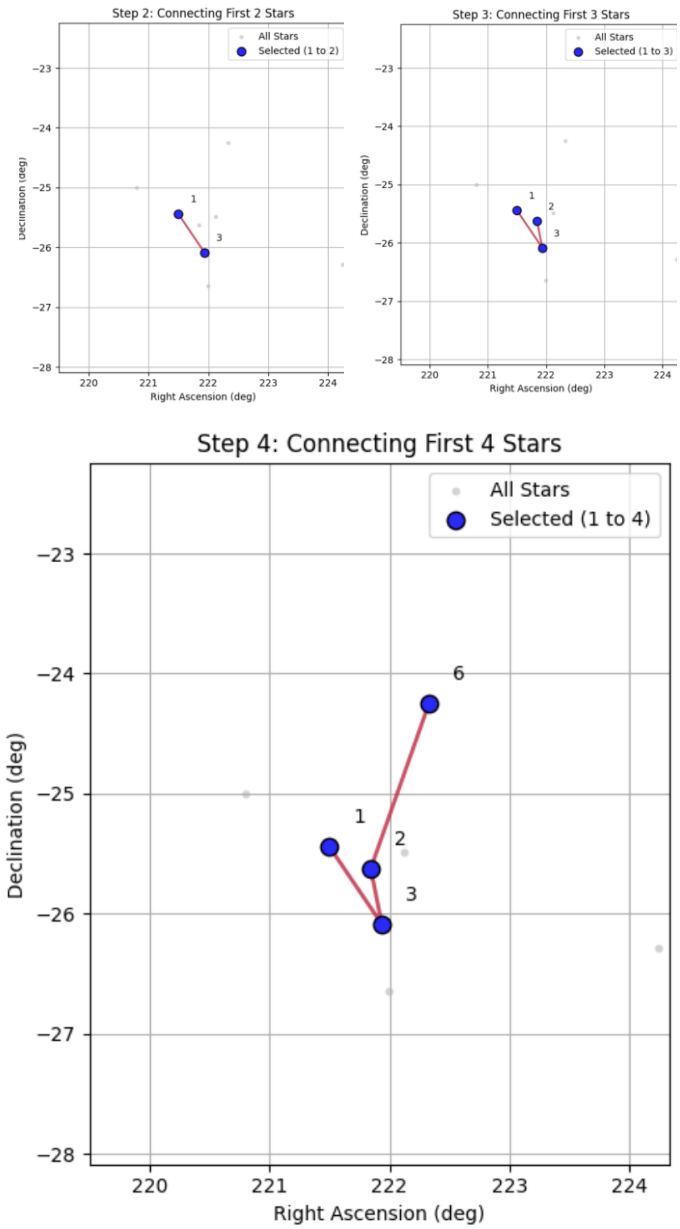


Figure 5. Final result of the

V. Result and Conclusion

The Best-First Branch and Bound (BFBnB) algorithm successfully identified an optimal subset of four stars forming a compact and well-distributed constellation with the best possible score under the given constraints. The selected stars IDs 1, 2, 3, and 6 achieved a final constellation score of -0.60, a significant improvement from earlier candidates, while maintaining a small spatial footprint of 1.52 units in area. This subset balanced magnitude visibility and angular arrangement, making it the most promising asterism from the initial pool of ten stars.

The algorithm's pruning strategy proved to be highly efficient, discarding numerous suboptimal branches based on upper-bound estimates and enforcing constraints such as maximum constellation area. Notably, Star 7 was consistently excluded due to violating the area constraint. This shows the effectiveness of integrating both geometric and heuristic criteria within a guided search, significantly narrowing the solution space without compromising optimality.

Overall, the experiment validates this method as a powerful tool for constrained combinatorial selection in astronomical contexts. Its ability to guarantee optimality while maintaining computational feasibility highlights its applicability in fields such as satellite navigation, constellation design, and automated star chart generation.

Future developments will focus on scaling the algorithm to handle larger star catalogs, potentially incorporating parallel processing to maintain tractability. Additionally, more sophisticated scoring functions could be introduced to capture photometric variability, color indices, or temporal stability for dynamic constellations. Integrating the method with real-time sky survey data and expanding it for multi-constellation optimization across different regions of the sky are also promising directions for applied astronomical research.

Video Link and Attachment

Video: https://youtu.be/ry0ir_TsBes

Github Link: github.com/najwakahanifatima/constellation-optimizer-bnb

Acknowledgement

The author would like to thank to God for the guidance throughout the process of learning and writing this paper. The author would also like to deliver biggest gratitude to IF2211 Strategi Algoritma lecturers for sharing and guiding the student in learning the materials throughout the semester. The author would also like to thank to family and friends who have accompanied the journey of joy and sorrow since the start of the author's university journey.

References

[1] [1] Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). Network Flows: Theory, Algorithms, and Applications. Prentice Hall.

[2] [2] Horowitz, E., Sahni, S., & Rajasekaran, S. (1998). Computer Algorithms. New York: Computer Science Press.

[3] [3] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press.

[4] [4] Kumar, V., & Kanal, L. N. (1983). A general branch and bound formulation for understanding and synthesizing and/or graph search strategies. Artificial Intelligence, 21(1–2), 179–198.

[5] [5] Lawler, E. L., & Wood, D. E. (1966). Branch-and-Bound Methods: A Survey. Operations Research, 14(4), 699–719.

[6] [6] Martello, S., & Toth, P. (1990). Knapsack Problems: Algorithms and Computer Implementations. Wiley.

[7] [7] Kelly, B. A., Kemp, C., Little, D. R., Hamacher, D., & Cropper, S. J. (2023). Visual perception principles in constellation creation. Topics in Cognitive Science, 15(1), 65–79. https://doi.org/10.1111/tops.12720

[8] [8] Liebe, C. C. (1995). Pattern Recognition of Star Constellations for Spacecraft Applications. Retrieved from https://backend.orbit.dtu.dk/ws/portalfiles/portal/3905478/Liebe.pdf

[9] [9] Chamberlain, C. (n.d.). Movable Type Scripts – Calculate Distance, Bearing and More Between Lat/Lon Points. Retrieved from https://www.movable-type.co.uk/scripts/latlong.html

[10] [10] Corral, M. (2013). The Law of Cosines. In Elementary Trigonometry. LibreTexts. Retrieved from https://math.libretexts.org/Bookshelves/Precalculus/Elementary_Trigonometry_(Corral)

[11] [11] Konivat. (n.d.). Hipparcos Star Catalog. Retrieved from https://www.kaggle.com/datasets/konivat/hipparcos-star-catalog

[12] [12] Kemp, C., Hamacher, D. W., Little, D. R., & Cropper, S. J. (2022). Perceptual grouping explains similarities in constellations across cultures. Psychological Science, 33(3), 354–363. https://doi.org/10.1177/09567976211044157

[13] Kemp, C., Hamacher, D. W., Little, D. R., & Cropper, S. J. (2022a). Comparing constellations across cultures. Nature Astronomy, 6, 406–409.

STATEMENT

I hereby declare that the paper I wrote is my own writing, not an adaptation, or translation of someone else's paper, and not plagiarized.

Bandung, 24 June 2025

Najwa Kahani Fatima - 13523043