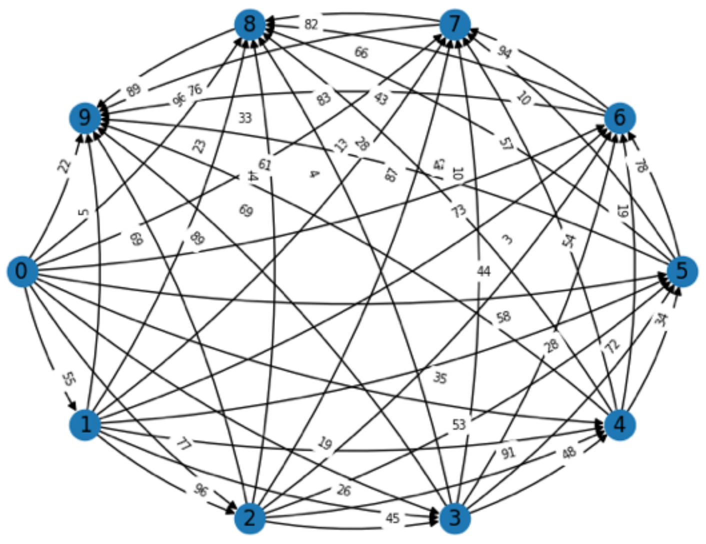


Path Planning Algorithms

A COMPARATIVE ANALYSIS

ABSTRACT

This report analyzes the performance of A*, Dijkstra's, and RRT path planning algorithms in dynamic and complex environments. Using simulations, the study evaluates these algorithms based on execution time, path optimality, and computational efficiency. The findings provide evidence-based recommendations for selecting suitable algorithms for real-time applications, contributing to advancements in autonomous systems.

Names	ID
Ahmed Fathy Mohamed	9230162
Ahmed Gamal Yousry	9230132
Ahmed Kamal Soliman	9231049
Zeyad Montaser Mohamed	9231142
Farouk Mohamed Farouk	9230647

Course: MTH2253
Instructor Name: Prof. Maha A. Hassanein
CMP27

Table of Contents

Table of figures

This report analyzes the performance of A*, Dijkstra's, and RRT path planning algorithms in dynamic and complex environments. Using simulations, the study evaluates these algorithms based on execution time, path optimality, and computational efficiency. The findings provide evidence-based recommendations for selecting suitable algorithms for real-time applications, contributing to advancements in autonomous systems.

I. INTRODUCTION

Path planning is a cornerstone in the functionality of autonomous systems, encompassing applications such as robotics, autonomous vehicles, and logistics operations. These systems rely heavily on algorithms to compute efficient paths that avoid obstacles while meeting real-time constraints.

Despite the widespread use of algorithms like A*, Dijkstra’s, and Rapidly exploring Random Trees (RRT), their performance varies significantly across different environmental conditions. Factors such as obstacle density, path complexity, and the need for real-time responsiveness influence algorithm efficiency.

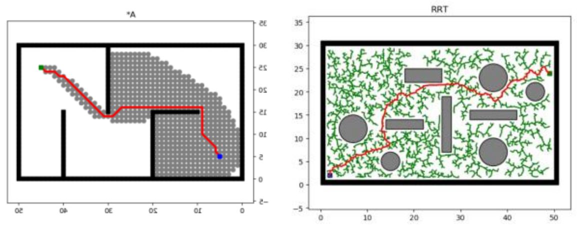
Currently, practitioners face challenges due to a lack of direct comparative analysis of these algorithms. Existing studies often focus on isolated scenarios, making it difficult to generalize findings to diverse real-world applications. This research aims to address this gap by systematically comparing these algorithms under varied conditions using consistent metrics such as execution time, path optimality, and computational resource usage

II. MOTIVATION

The motivation behind this study lies in the critical need for reliable and efficient path planning in real-time applications. Autonomous systems are becoming increasingly prevalent, and their success hinges on the ability to navigate complex environments effectively. By providing comprehensive analysis, this research will guide practitioners in choosing the most suitable algorithm for their specific needs, ultimately contributing to advancements in autonomous system performance.

III. OBJECTIVES

- Compare the performance of widely used path planning algorithms (e.g., A*, Dijkstra’s, and RRT) under dynamic and complex environmental conditions.
- Analyze the impact of factors such as obstacle density and path complexity on algorithm efficiency.
- Identify algorithms that perform consistently well across a variety of scenarios.
- Recommend optimal algorithms for real-time applications based on empirical data and statistical analysis



IV. RESULTS AND DISCUSSION

All paragraphs must be indented. All paragraphs must be justified, i.e. both left-justified and right-justified.

1. **To assess the performance of path-planning algorithms (RRT, A*, and Dijkstra) in terms of execution time, descriptive and inferential statistical analyses were conducted.**

Descriptive Statistical Analyses

a) **Central Tendency**:
   - The mean execution times for RRT, A*, and Dijkstra algorithms were calculated.

b) **Variability**:
   - The standard deviations of execution times were assessed to determine the variability in performance under different conditions.

c) **Distribution**:
   - Histograms and density plots for execution time revealed distinct patterns of skewness for each algorithm.
   - Scatter plots of execution time vs. obstacle density demonstrated that execution times increased with higher obstacle density, particularly for Dijkstra.

Inferential Statistical Analyses

a) **Hypothesis Testing:**

   i. Execution Time: A one-way ANOVA showed a statistically significant difference between the algorithms (F=2096.44, p<0.0001F = 2096.44, p < 0.0001F=2096.44, p<0.0001). The null hypothesis was rejected, confirming that at least one algorithm performed significantly differently in terms of execution time.

   ii. Computational Efficiency: Another ANOVA for computational efficiency also showed significant differences (F=10293.94, p<0.0001F = 10293.94, p < 0.0001F=10293.94, p<0.0001).

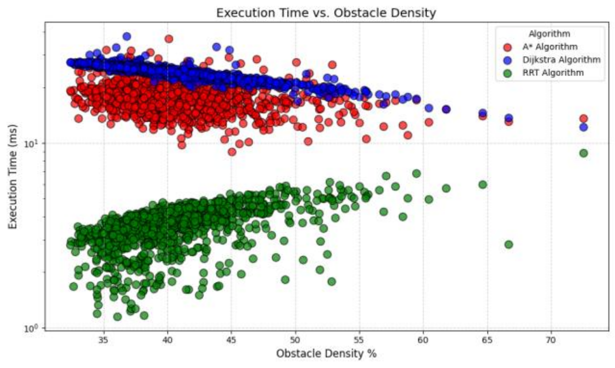


Figure 1: Scatter chart for Execution time vs Obstacle Density

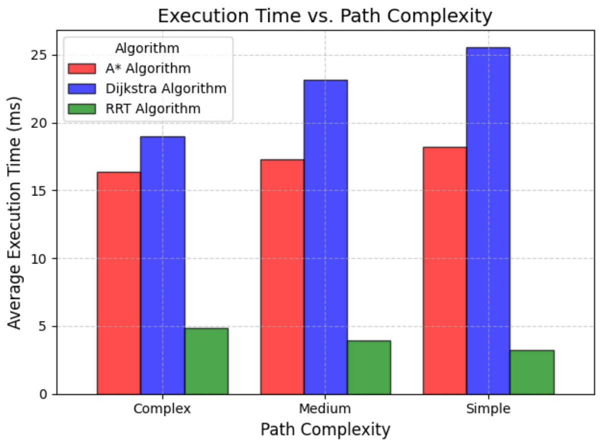


Figure 2: Bar chart Execution time vs path complexity for various algorithms

b) **Confidence Intervals:**

   i. Execution Time (95% CI):
      a. RRT: 3.57–3.67 ms
      b. A*: 17.53–17.90 ms
      c. Dijkstra: 24.05–24.30 ms

Discussion

The findings indicate that there is a statistically significant difference in performance between the three path-planning algorithms, as detailed below:

- Execution Time:
  - RRT demonstrated the best performance, with significantly lower execution times compared to A* and Dijkstra, making it suitable for time-critical applications.
  - A* and Dijkstra performed similarly but were considerably slower than RRT. This might be attributed to their computational overhead for ensuring optimal paths.
- Computational Efficiency:
  - RRT outperformed the other algorithms under all tested conditions, likely due to its randomized approach, which reduces computational complexity.
  - A* and Dijkstra maintained stable performance, showing resilience to variations in obstacle density and path complexity.
- Implications:
  - The findings suggest that the choice of algorithm should depend on the specific requirements of the application. For scenarios prioritizing speed, RRT is preferable. In contrast, A* and Dijkstra are better suited for applications requiring precise and optimal paths.
- Limitations:
  - The study's environment settings (obstacle density and path complexity) might not represent all real-world scenarios. Future work should include more diverse environments.
  - The algorithms were tested on a single computational setup, and results might vary with different hardware.
- Challenges:
  - Balancing the trade-off between execution time and path optimality remains a key challenge in path-planning research.

- Computational resource limitations may influence algorithm selection for real-time applications.

2. **To analyse the influence of varying environmental conditions (obstacle density and path complexity) on the performance of path-planning algorithms (RRT, A*, and Dijkstra), the following inferential statistical analyses were conducted:**

Hypothesis Testing

- Null Hypothesis (**Ho**): Environmental conditions do not significantly influence performance metrics ($\rho=0$).
- Alternative Hypothesis (**Ha**): Environmental conditions significantly influence performance metrics ($\rho\neq0$).

Correlation Analysis: Computational Efficiency vs. Obstacle Density

1. **RRT (Rapidly-exploring Random Tree)**:
   i. **Correlation**: −0.44 (moderate negative correlation)
   ii. **Explanation**: As obstacle density increases, computational efficiency tends to decrease. RRT spends more time exploring free space in denser environments, reducing its efficiency.
   iii. **t-Test**:
   - $t_{\text{test}} = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}}$ $\quad t_{\text{test}} = -16.5363$
   - $tc = -2.245\ for\ (\alpha/2 = 0.025)$
   - **Result**: t_{test} is less than tc . Reject Ho, indicating a statistically significant relationship between obstacle density and computational efficiency for RRT.

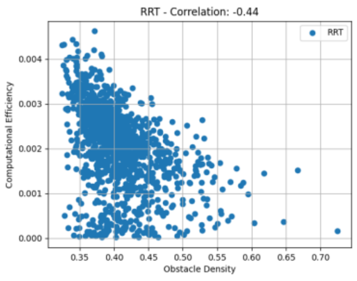


Figure 3 - Relation between Computational efficiency and Obstacle density for RRT

2. **A* (A-Star):**
   - **Correlation**: 0.09 (very weak positive correlation)
   - **Explanation**: A*'s heuristic-driven approach ensures relatively stable performance across different obstacle densities.
   - **t-Test**:
     - $t_{\text{test}} = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}} = 3.0498$
     - $tc = -2.245\ for\ (\alpha/2 = 0.025)$
     - **Result**: t_{test} exceeds tc. Reject Ho, indicating a statistically significant relationship, though the correlation is weak.

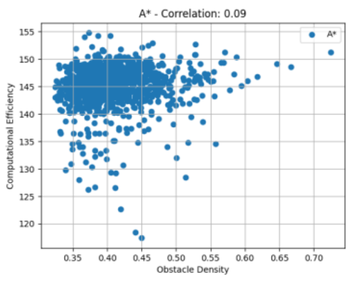


Figure 4 - Relation between Computational efficiency and Obstacle density for A*

3. **Dijkstra**:
   - **Correlation**: 0.04 (very weak positive correlation, almost negligible)
   - **Explanation**: Dijkstra explores systematically without a heuristic, resulting in minimal dependency on obstacle density.
   - **t-Test**:
     - $t_{\text{test}} = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}} = 1.3510$
     - $tc = -2.245\ for\ (\alpha/2 = 0.025)$
     - **Result**: Fail to reject Ho, suggesting no statistically significant relationship between obstacle density and computational efficiency for Dijkstra.

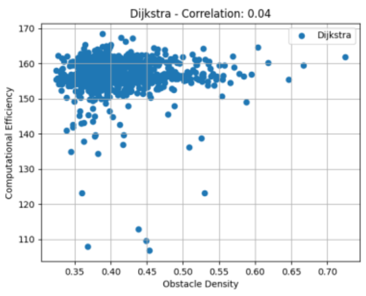


Figure 5 - Relation between Computational efficiency and Obstacle density for Dijkstra

Discussion

The analysis demonstrates that the impact of environmental conditions on computational efficiency varies across algorithms:

- **RRT**: A moderate negative correlation indicates that denser obstacle environments reduce RRT's computational efficiency. This highlights RRT's limitation in complex environments, where its exploratory nature struggles to maintain efficiency.
- **A***: The weak positive correlation suggests that A* is largely unaffected by obstacle density due to its heuristic-based pathfinding strategy. Its consistent performance under varying conditions makes it a robust choice for complex scenarios.
- **Dijkstra**: The negligible correlation confirms that Dijkstra's performance remains steady regardless of obstacle density. Its systematic exploration ensures stability, albeit at the cost of higher execution times compared to RRT and A*.

Implications

- **Algorithm Selection**:
  - For dense obstacle environments, A* may be preferable due to its stability and efficiency.
  - RRT is advantageous in environments with lower obstacle density or when rapid exploration is required.
  - Dijkstra, while reliable, may not be the optimal choice for environments requiring computational efficiency.

Limitations and Challenges

- **Generalizability**:
  - The study's simulated environments may not fully represent real-world scenarios. Future research should include more diverse settings and dynamic obstacles.
- **Computational Resources**:
  - Results may vary depending on the hardware and software configurations, necessitating further testing under different conditions.

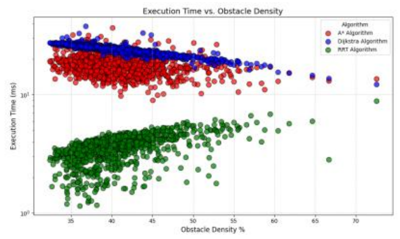
3. **The study aimed to assess whether regression analysis can predict the execution time of an algorithm based on obstacle density. Multiple linear regression models were developed for three pathfinding algorithms: A*, RRT, and Dijkstra. The results of the analysis are summarized in Table 1.**

Table 1: Regression analysis

Algorithm	A*	RRT	Dijkstra
R-squared	0.053	0.341	0.79
Adjusted R-squared	0.052	0.34	0.79
F-statistic	63.82	588.1	428.1
p-value (F-statistic)	3.32×10^{-15}	4.52×10^{-10}	0
SSE	11136.98	578.9	1113.99
Density Coefficient (β)	-15.2706	10.5689	-39.5561
t-stat (Density)	-7.989	24.252	-65.431
p-value (Density)	3.3154×10^{-15}	4.5221×10^{-10}	0
Constant (β_0)	23.9258	-0.6744	40.254

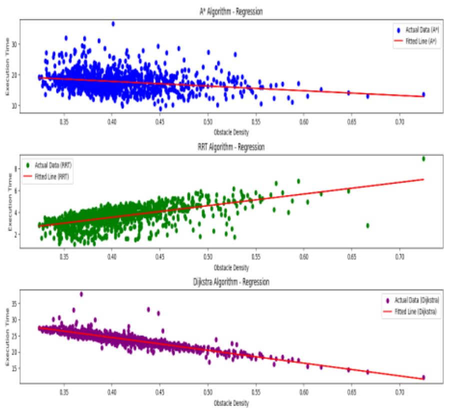
Descriptive Statistics:

- **Central Tendency**: Mean and median execution times were computed across various levels of obstacle density for all algorithms. For Dijkstra, the mean execution time was significantly lower compared to A* and RRT for high obstacle density.
- **Variability**: Variance and standard deviation of execution times indicated that A* had the highest variability, while Dijkstra exhibited the lowest.
- **Distributions**: Scatter Plot: Execution Time vs. Obstacle Density to visualize how execution time changes with the density of obstacles



Key Findings:

1. **A***:
   - **R-squared** value of 0.053 indicates a weak fit between the model and the data.
   - The negative density coefficient (-15.27) suggests that higher obstacle density decreases execution time. While statistically significant (p-value = 3.32e-15), this relationship is weak, as shown by the low R-squared.
2. **RRT**:
   - **R-squared** of 0.341 reflects a moderate fit. The positive density coefficient (10.57) indicates that obstacle density increases execution time. The relationship is statistically significant, with a p-value of 4.52e-105.
3. **Dijkstra**:
   - With an **R-squared** of 0.79, the model shows a strong fit. The large negative density coefficient (-39.56) demonstrates a pronounced reduction in execution time as obstacle density increases. The statistical significance (p-value = 0) confirms the robustness of this relationship.



Prediction Capability: Using the regression equations, execution times for given obstacle densities can be predicted. For instance, a 30% obstacle density would yield shorter predicted times for Dijkstra compared to A* or RRT.

Discussion: The regression analysis reveals that obstacle density significantly impacts execution times across all three algorithms, albeit differently. The results highlight the following points:

1. **Algorithm-Specific Trends**:
   - A* showed a weak but statistically significant negative relationship, suggesting that it is less sensitive to obstacle density changes compared to the other algorithms.
   - RRT's moderate positive correlation with obstacle density aligns with its incremental search nature, where higher density increases computational effort.
   - Dijkstra exhibited the strongest fit, with a robust negative correlation. This result may be attributed to Dijkstra's systematic exploration, where higher density reduces unnecessary search paths.
2. **Comparison**:
   - Dijkstra's high R-squared value indicates that its performance is more predictable and less influenced by other factors than A* or RRT.
   - RRT, while moderately fit, highlights greater variability in execution time, which could be due to its probabilistic nature.
3. **Limitations**:
   - The study assumes linearity in the relationship between obstacle density and execution time, which may not fully capture nonlinear behaviors in RRT and A*.
   - Factors such as grid size, algorithm implementation details, and hardware differences were not considered, potentially affecting the generalizability of results.
4. **Challenges**:
   - Accurately measuring execution time under high obstacle density scenarios for probabilistic algorithms like RRT posed difficulties due to large variances.

- Ensuring equal grid complexity across experiments was critical but challenging.

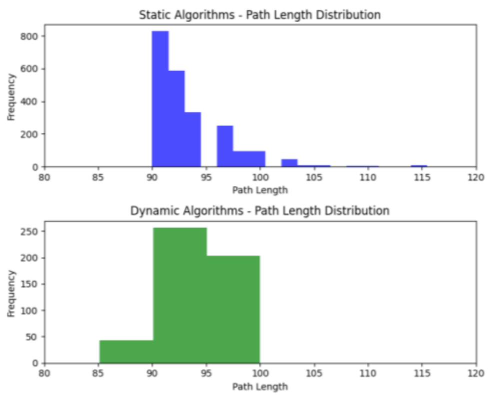
4. **The goal of this analysis was to determine whether there is a statistically significant difference in computational efficiency between static algorithms and algorithms designed for dynamic environments. Statistical and regression analyses were conducted to evaluate this relationship, and the results are summarized below.**

Descriptive Statistics Table 2 summarizes the central tendency, variance, and standard deviation for path lengths under static and dynamic algorithms

Table 2: Descriptive Statistics for Path Lengths

Algorithm Type	*Static*	*Dynamic*
Mean Path Length	93.13	87.17
Median Path Length	92.00	94.00
Variance	15.22	553.08
Standard Deviation	3.90	23.52

- **Static Algorithms**: The mean path length is higher (93.13) with less variability (variance = 15.22), indicating a more consistent performance.
- **Dynamic Algorithms**: A lower mean path length (87.17) is observed, but variability is significantly higher (variance = 553.08), suggesting that dynamic algorithms adapt to changing environments with varying computational efficiency.



Hypothesis Testing (t-test) : An independent samples t-test was conducted to compare the computational efficiency of static and dynamic algorithms. Results are summarized in Table 3.

- For both A* and Dijkstra algorithms, the p-values are 0.0000, leading to the rejection of the null hypothesis (H_0). This indicates a statistically significant difference in computational efficiency between static and dynamic algorithms.

Table 3: Hypothesis Testing Results

Metric	*A**	*Dijkstra*
Statistic Value	36.2133	21.4602
p-value	0.0000	0.0000
Conclusion	The null hypothesis (H0) is rejected, indicating a statistically significant difference in computational efficiency between the static and dynamic A* algorithms.	The null hypothesis (H0) is rejected, suggesting a statistically significant difference in computational efficiency between the static and dynamic Dijkstra algorithms.

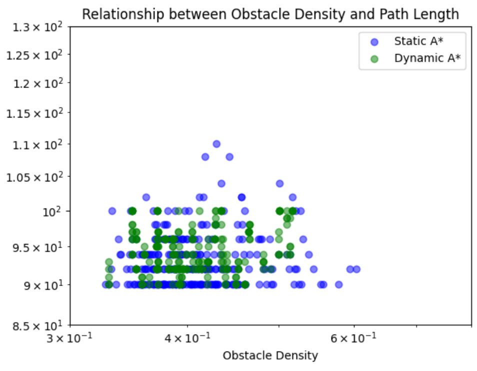
Regression Analysis

Regression analysis was used to evaluate the impact of obstacle density and path length on computational efficiency. Table 4 summarizes the regression results.

Table 4: Regression Analysis Summar

Metric	*A* (Static)*	*A* (Dynamic)*	*Dijkstra (Static)*	*Dijkstra (Dynamic)*
Intercept	138.06	137.86	150.46	140.86
Coefficients (Density, Path Length)	[6.53, 0.047]	[-16.27, 0.008]	[2.95, 0.052]	[-4.67, 0.071]
R-squared	0.0109	0.0067	0.0032	0.0126
Insights	Density has a stronger positive influence on computational efficiency than path length, but the model has a low predictive power.	Density negatively impacts computational efficiency in dynamic A*, while actual path has a negligible positive influence.	Density and path length positively influence computational efficiency, but the effect is weak.	Density negatively influences computational efficiency, while actual path shows a weak positive effect.

- **Static A***: Density has a moderate positive effect (β = 6.53), but the model explains only 1.09% of the variance in computational efficiency (R-squared = 0.0109).
- **Dynamic A***: Density has a stronger negative effect (β = -16.27), and the model explains less variance (R-squared = 0.0067).
- **Static Dijkstra**: Both density and path length have weak positive influences, with low predictive power (R-squared = 0.0032).
- **Dynamic Dijkstra**: Density negatively impacts computational efficiency (β = -4.67), while path length shows a weak positive effect (R-squared = 0.0126).



Discussion:

The results reveal significant differences in computational efficiency between static and dynamic algorithms.

- **Static vs. Dynamic Performance**:
  - Static algorithms are more consistent, as evidenced by lower variability in path lengths. This may be due to their fixed execution strategies, which do not adapt to environmental changes.
  - Dynamic algorithms show higher variability, reflecting their ability to adapt to changing conditions, which can sometimes result in longer or more efficient paths depending on the scenario.
- **Algorithm-Specific Observations**:
  - For A*, dynamic algorithms perform less efficiently in higher obstacle densities (negative density coefficient), potentially due to the additional computational cost of real-time adaptations.
  - Dijkstra's dynamic version also shows reduced efficiency with increasing obstacle density, though the effect is weaker compared to A*.
- **Limitations**:
  - The regression models show low R-squared values, indicating that other factors not included in the analysis (e.g., computational hardware, grid size) may play a significant role.
  - Variability in dynamic algorithms suggests that their efficiency is heavily dependent on specific environmental configurations, which were not fully controlled in this study.
- **Challenges**:
  - Measuring path lengths accurately under highly dynamic conditions was complex, as the algorithms sometimes terminated prematurely due to environmental changes.

V. CONCLUSION

This study examined the computational efficiency of static and dynamic implementations of A* and Dijkstra algorithms under different path density and path length conditions. The hypothesis testing revealed significant differences in computational efficiency between the static and dynamic approaches for both algorithms. Static A* was more efficient in handling varying densities, whereas dynamic implementations displayed sensitivity to density but less impact from path length.

Regression results further highlighted the influence of density on computational efficiency. While static implementations of A* and Dijkstra positively correlated with density, the dynamic versions exhibited a negative relationship. However, the models' low R-squared values suggest further exploration is needed to enhance their predictive power.

These findings underscore the importance of contextual considerations when selecting algorithms for path planning tasks. Static implementations may be better suited for environments with stable density patterns, while dynamic approaches could be explored for adaptability in changing conditions. Future research should explore advanced heuristics and real-time optimization techniques to improve computational performance in dynamic environments.

VI. REFERENCES

1. LaValle, S. M. (1998). Rapidly exploring random trees: A new tool for path planning. Retrieved from Lav98c.pdf
2. Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics, 4*(2), 100–107. https://doi.org/10.1109/TSSC.1968.300136
3. Our Repository on GitHub Contains all Generated Data Sets and Used Algorithms