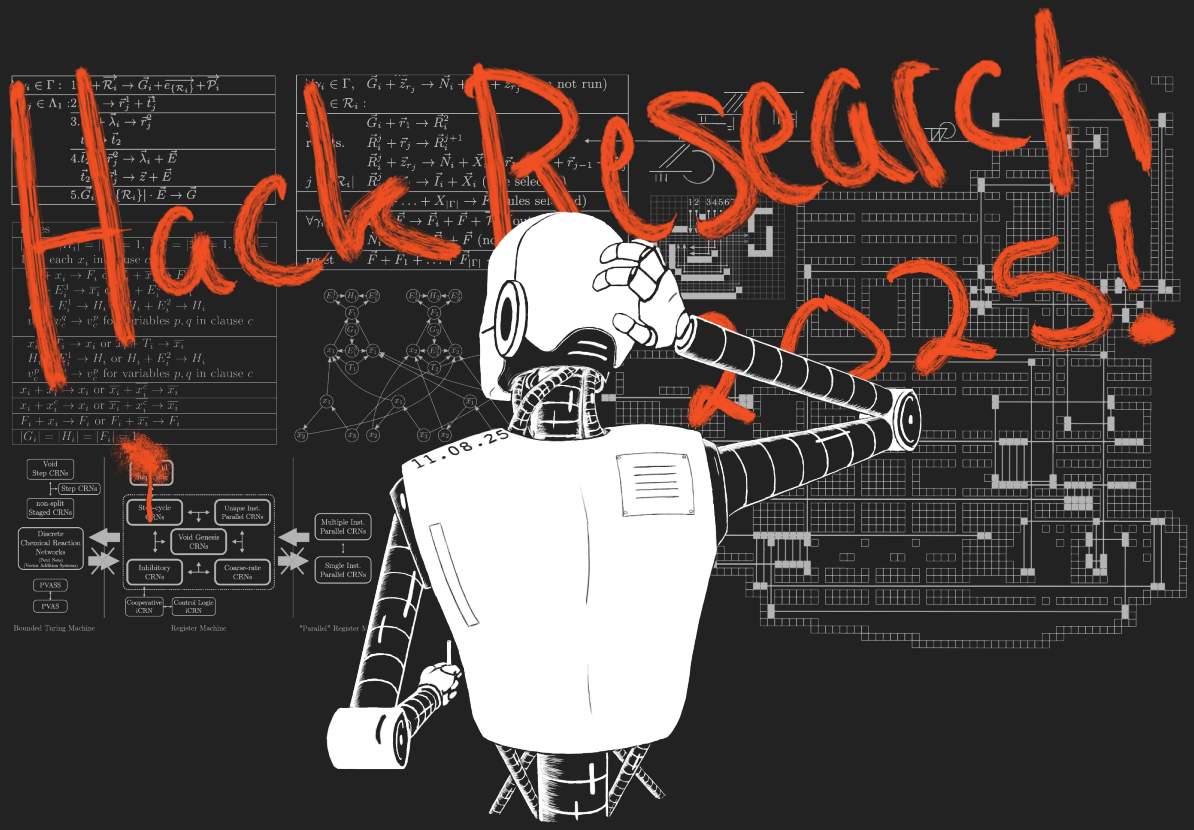


Proceedings of the 6^{th} Hack Research Hackathon
University of Texas Rio Grande Valley

$\mathcal{H}ack\mathbb{R}$ 2025

Tim Wylie, Editor

Copyright © 2025 for the individual papers by the papers' authors. Copying permitted only for private and academic purposes. This volume is published and copyrighted by its editors.

Preface

The event this year continued to grow and change in some remarkable ways. We were at the same venue for the third year, and the overall process continues to be refined with better efficiency and more support and resources from the university and outside sponsors.

We continued having breakout rooms, a lot of SWAG, had cleaner registration, continued the tradition of games and an art wall, changed up the meals a bit, and simplified the awards and prizes. We again had substantial participation from the faculty and growing engagement from graduate students. In total, we had about 100 students attend with 18 submitted entries and a lot of research that didn't quite make it to a finished paper.

Overall, the event continues to evolve to meet the needs of our student body, the faculty, and our department. As we advance the graduate programs, hopefully this will also progress the event as a useful way to explore new topics and problems and to connect with faculty and their research in a worthwhile manner.

November 8-9, 2025
Edinburg, TX, United States

Tim Wylie, Editor

Acknowledgements

This event, surprisingly, doesn't ever seem to get easier to put together and run. That is, at least on the whole. My part, however, grows easier every year by more and more help from everyone else. I can not express my gratefulness to the support system that I rely on.

The department has embraced the event in many ways. Odette Perez and Laura Cantu took care of the reservations, meals, snacks, and any number of miscellaneous things that were needed. Of course the faculty provided problems, but their main support came from the time they dedicated to attend the event, encourage the students, and help them throughout the process.

We were fortunate to have funding from an NSF research grant, the Google exploreCSR program, and through the Julia Beecherl Endowment (via Dr. Robert Schweller).

I'm thankful to the students that came and struggled through their weekend, which made the event possible and successful. The dedication to learn and endure in a stressful environment demonstrates a lot of commitment and tenacity.

Finally, I'm always indebted to my family, their flexibility and understanding, and their support.

Sponsors



www.utrgv.edu/csci



www.nsf.gov/
National Science Foundation Grant CCF-2329918.



research.google/outreach/explorecsr/

Support was also provided as part of the Julia Beecherl Endowment through Dr. Robert Schweller.

Organization

$\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025 was organized by the Algorithmic Self-Assembly Research Group (ASARG) with the help of the Department of Computer Science at the University of Texas Rio Grande Valley. The event was graciously hosted by UTRGV and funded by several sources listed in the Acknowledgements.

The Program Committee is comprised of the faculty that judged the submissions. Those faculty also submitted problems along with the faculty listed in Volunteers. The ASARG members are default volunteers.

Director

Tim Wylie	timothy.wylie@utrgv.edu	U. of Texas Rio Grande Valley

Program Committee

Bin Fu	bin.fu@utrgv.edu	U. of Texas Rio Grande Valley
Yifeng Gao	yifeng.gao@utrgv.edu	U. of Texas Rio Grande Valley
Austin Luchsinger	austin.luchsinger@utrgv.edu	U. of Texas Rio Grande Valley
Robert Schweller	robert.schweller@utrgv.edu	U. of Texas Rio Grande Valley
Emmett Tomai	emmett.tomai@utrgv.edu	U. of Texas Rio Grande Valley
Li Zhang	li.zhang@utrgv.edu	U. of Texas Rio Grande Valley

Volunteers

Marzieh Ayati
Laura Cantu
Sergei Chuprov
Erik Enriquez
Qi Lu
Odette Perez

ASARG Members

Alberto Avila-Jimenez
Divya Bajaj
David Barreda
Jose Luis Castellanos
Ryan Knobel
Aiden Massie
Pablo Santos
Ramiro Santos

Results

Winners

The awards were given to the top submission based on different research categories. Prizes were given based on the categories of theory, data mining, algorithms, and robotics. Sixteen prizes were awarded, so additional prizes were awarded based on overall quality, team size, and the number of awards. The award winners were:

- Outlier Detection in Espresso Brewing
  Mario Camarena, Oziel Sauceda, Samantha Ocaña
- Creating Binary Circuits Through Tile Tumbling Burnout Bots
  Ricardo Balvanera, Nathaniel DeLeon, Jesus Gutierrez, Joanna Gutierrez, Carlos Torres Espinoza
- Coffee Brewing Outlier
  Marcelo Cavazos, Silvia Flores-Osuna, Dario Hinojosa
- Hybrid-Phase Foraging: A Scalable Algorithm for Complete Resource Collection in Robot Swarms
  Nathaniel Salazar, Luis De la Mora
- Burning out is easy. Specially at HackR
  Alberto Avila Jimenez, Pablo Santos
- 1-Configuration in Inhibitory Surface Chemical Reaction Networks on Grids is NP-hard
  Aiden Massie
- Optimal Vocabulary Construction for Subword Tokenization
  Gaukhar Nurbek
- A Greedy Approximation Algorithm for Optimal Vocabulary Construction in Subword Tokenization
  Damian Villarreal

Table of Contents

Mechanical Computation with Roons 1
Jonathan Navarro, Dustin Boston, Miguel Benavides . . . . . .

Outlier Detection in Espresso Brewing 3
Camarena Mario, Sauceda Oziel, Ocaña Samantha . . . . . .

Creating Binary Circuits Through Tile Tumbling Burnout Bots 6
Ricardo Balvanera, Nathaniel DeLeon, Jesus Gutierrez, Joanna Gutierrez, Carlos Torres Espinoza . . . . . .

Finding the Outlier Espresso Using A Data-Driven Approach to Brewing Anomalies 9
Kacey Starr, David Gonzalez, Luis Lozano . . . . . .

Coffee Brewing Outlier 11
Marcelo Cavazos, Silvia Flores-Osuna, Dario Hinojos . . . . . .

Hybrid-Phase Foraging: A Scalable Algorithm for Complete Resource Collection in Robot Swarms 14
Nathaniel Salazar, Luis De la Mora . . . . . .

Burning out is easy. Specially at HackR 16
Alberto Avila Jimenez, Pablo Santos . . . . . .

Thermodynamic Affinity Networks can cycle 18
Alberto Avila Jimenez, Ramiro Santos . . . . . .

1-Configuration in Inhibitory Surface Chemical Reaction Networks on Grids is NP-hard 21
Aiden Massie . . . . . .

Optimal Vocabulary Construction for Subword Tokenization 23
Gaukhar Nurbek . . . . . .

Problems in small circles and big cycles 25
Jose Luis Castellanos, Pablo Santos . . . . . .

Coffee Brewing Outlier 28
Fernando Vela, Sean Solis . . . . . .

PhonoDetect: Cross-Language Phoneme Analysis and Live Speech Transcription 30
Alissen D. Moreno . . . . . .

Increasing Efficiency in Late Stage Search with Swarm Robotics 33

Jose Amaro, Mark Hernandez, Koriel Lopez . . . . . .

Anomaly Detection in Espresso Extraction Profiles: A Data Mining Approach to Coffee Quality Control **36**
Damian Villarreal . . . . . .

Comparative Analysis of Byzantine-Robust Aggregation Strategies in Federated Learning Under Label Flipping Attacks **41**
Damian Villarreal . . . . . .

A Greedy Approximation Algorithm for Optimal Vocabulary Construction in Subword Tokenization **44**
Damian Villarreal . . . . . .

Gin Rummy with Perfect Information **49**
Alenis Chavarria, Ryan Knobel . . . . . .

Mechanical Computation with Roons

Jonathan Navarro * Dustin Boston † Miguel Benavides ‡

Abstract

Roons is an upcoming marble-based mechanical computation system designed by Whomtech that demonstrates the physical implementations of logical and computational processes.This paper explores how Roons can simulate a Turing Machine (TM), thereby demonstrating universal computation. Using a small example TM, we show the correspondence between symbolic computation and the mechanical structure of the Roons.

1 Introduction

Mechanical computation systems are used to demonstrate how physical devices, such as Roons**,** can perform algorithmic processes like computers that we have today. One of the examples of these mechanical systems is Roons, which is a kickstarter platform built on marbles designed by Whomtech. Similar to other mechanical computers such as the Digi-Comp and Difference Engine, Roons uses marbles that can represent data and perform logic operations like AND, OR, XOR, NOT through a network of paths also called Roons. The goal of this report is to explore how Roons can be used to simulate a Turing Machine (TM) by demonstrating that the system supports universal computation. This report aims to discuss and define how a simple Turing Machine could be implemented using Roons.

1.1 Capabilities of Roons

Roons is a marble-based mechanical computing system and is capable of performing both digital and logical computation by combining AND, OR, XOR, and NOT. Theoretically, it is: Turing universal on an unbounded board, PSPACE-complete when limited to a finite area, and Turing-complete: performs any computation that a computer can, given enough time and memory. Whomtech's website additionally provides us with documentation and useful tutorials showing how to construct XOR gates, binary adders, registers, and even a basic calculator. These components mirror the building blocks of modern digital systems that we use today.

2 Our Results

Noticing the marbles were on a two-turn basis where the first turn corresponded to every odd row and the next turn to every even row, we discovered that we could manipulate the marbles to remain in the same row for an extra turn (effectively turning a 1 into a 0). We also found that we could position the marbles midway between 0 and 1, effectively creating a state that could be represented as 0 and 1 simultaneously. This intermediate state could then be resolved when another marble acted on it, pushing it definitively to either 0 or 1. These interactions introduced model controlled transitions between binary states and became major when exploring more complex logic.

Next, we discovered a method to store memory by creating a looping interval closed path where an outside marble's interaction (or lack thereof) determined whether data could be read or written. This enabled us to store information such as our current state as well as to choose when to read from or write to that information. We can now take our new stored values of the current state and string to pass into a Decision mechanism, which chose which logic to execute based on information passed through. The logic itself was implemented using components such as switches, canutes, distributors, and traps, which allowed manipulation of marble flow to achieve a desired outcome. The outcome produced by the system was an updated state , an updated string, and a direction (left or right). The new state and string were stored in their respective memory locations, and the data was written to the string before initiating a new loop cycle.

Finally, we determined a way to synchronize outputs to inputs by ensuring that information was received and processed at the correct time within each interval. We solved this using a global clock signal combined with signals from each seperate process. Each process outputs a "READY" flag when it finishes its task. These signals are then combined through a synchronizer gate that produces a single signal once all processes have completed. This unified signal triggers the machine's next turn. When the new turn begins, all processes reset their READY flags, allowing the cycle to repeat in synchronization.

2.1 Final Result

Based on our findings, we demonstrated that Roons satisfy all the basic characteristics of a Turing Machine. The manipulation of marbles in the binary states of (0 and 1) was established as a mechanism for symbol representation and manipulation. The intermediate logic state enabled condition transitions between binary values to represent computational decision-making. Our looping interval path represented a memory system equivalent to the Turing tape in its ability to retain and retrieve data. The decider enacted the transition function through the determination of next state, output, and complementary

*Department of Computer Science, University of Texas Rio Grande Valley, jonathan.navarro02@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, dustin.boston01@utrgv.edu

‡Department of Computer Science, University of Texas Rio Grande Valley, miguel.benavides02@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2024. It should be considered a workshop preprint.

directional movement (left or right), based on its inputs at that moment. Finally, as a mechanism to tie these together, we also synchronized actions using a global clock and READY protocol signs to indicate the order of action to accomplish step by step actions, likewise to Turing's description of a Turing Machine. In combination, we showed that Roons certainly operated as a mechanical realization of a Turing Machine, capable of controlled, sequential computation.

3 Turing Machine Implementation

The example Turing Machine (TM) we used for this project consists of four states (q_1, q_2, q_3, and A) and operates over a binary alphabet $\{0, 1\}$. The language for this TM is $L = \{\, w \subseteq \{0,1\}^* \mid w \text{ starts with 11 or 00} \,\}$. Its transition rules are shown in Table:

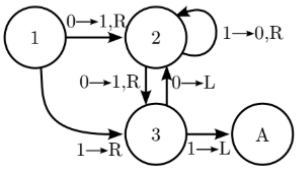


Figure 1: Turing Machine diagram.

State	Read	Write	Move	Next State
q_1	0	1	R	q_2
q_1	1	1	R	q_3
q_2	1	0	R	q_2
q_2	0	0	L	q_3
q_3	1	1	L	A

Table 1: Transition table for the example Turing Machine.

3.1 Mapping to Roons

In the Roons system logic:

- The **tape** is represented by a single row of marbles, where each position is either a 0 (No marble) or a 1 (marble present). This way we are able to implement our string.
- The **head** is a movable mechanism that will read and write symbols onto the the tape. At each step, it will detect the current state of the marble (0 or 1), then it will update the input according to the state instructions, and then shift one position to the left or right.
- Each **state** is implemented as a region of gates that directs the head's instructions depending on the current symbol being read by the **head**.

3.2 How transitions correspond to mechanical actions:

1. Reading a 0 or 1 is detected by the **head** mechanism.
2. Writing occurs by placing or removing a marble on the string.
3. Moving left or right changes the direction of the marbles along the tape.

The simulation was conceptualized using Whomtech's online **Roons** simulator, which allows custom **Roon** marble-based circuits.

3.3 Dynamic Memory Implementation

Roons is able to replicate dynamic memory by utilizing selectors and multiple branching paths. Each tape cell has a number of branching paths that guide the head towards the associated position depending on the current state and the encoded symbol read from the tape cell. Selectors at each of the junctions forward the head towards the correct tape position for reading or modifying the symbol as instructed by the state instructions. After the head modifies a tape cell, it proceeds according to the current state and transition rule to reach the next cell, following the path of the instruction. This is how Roons can simulate dynamic memory by allowing the head and state control logic to access and modify the tape arbitrarily during computation.

4 Conclusion

Roons provides us a concrete framework for understanding computation through physical mechanical systems. The current evidence provided demonstrates how to implement a Turing Machine using marbles, paths, and gates to show that Roons supports universal computation and is Turing-complete. Like other mechanical computers such as the Digi-Comp and Difference Engine, Roons realizes algorithmic processes physically as interactions of marbles, paths, and gates. Roons could be a successful educational and research tool for understanding computation in a concrete, tactile way.

References

[1] T. Neary and D. Woods. Four small universal turing machines. *Fundamenta Informaticae*, 91(1):123–144, 2009.

[2] Whomtech. Roons: Craft your own computers. `https://whomtech.com/`, 2025. Accessed 2025-11-09.

[1] [2]

Outlier Detection in Espresso Brewing

Camarena Mario * Sauceda Oziel † Ocaña Samantha ‡

Abstract

Espresso brewing is a precise process influenced by variables such as temperature, pressure, flow rate, and bean freshness. As these parameters fluctuate, they directly impact cup quality, creating a need for quantitative methods to ensure consistency and detect defective brews. Accordingly, this paper presents a data-mining model to identify anomalous espresso shots. Thirteen brewing logs recorded from a DE1 espresso machine were analyzed, each capturing the time-dependent behavior of pressure, flow, temperature, and water dispensed. The data were temporally aligned, standardized, and summarized using statistical features such as mean, maximum, and cumulative values. An L2-norm distance metric was then applied to compute an outlier score for each shot. The analysis successfully identified one shot that deviated across multiple parameters, particularly in pressure and flow profiles. The findings show that simple distance-based analysis, when combined with normalization and feature extraction, can reliably detect brewing inconsistencies and support data-driven quality control in espresso preparation.

1 Introduction

Espresso brewing is a precise process influenced by temperature, pressure, flow rate, and bean freshness. Small variations in these parameters can change how water moves through the compacted coffee bed, affecting extraction yield and the concentration of dissolved compounds in the final cup. The resulting differences are part of the extraction kinetics that define espresso quality. Maintaining a consistent cup requires adjusting grind size, dose, and temperature daily, a practice known as dialing in. Yet, even under controlled conditions, environmental factors such as humidity, bean aging, and machine variation continue to introduce variability.

Apiletti and Pastor [1] showed that espresso quality depends not only on mechanical parameters such as pressure and temperature but also on environmental conditions. They observed that humidity can affect grind behavior, requiring coarser settings in humid environments and finer ones in dry air to maintain the same extraction flow. This finding emphasizes that consistency in espresso preparation involves adapting to both machine conditions and surrounding climate.Similarly, Schmieder et al. [3] demonstrated that changes in flow rate, grind level, and temperature significantly influence extraction kinetics, with finer grinds producing higher resistance and pressure. Their study using a DE1 espresso machine revealed how physical variables interact during brewing, underscoring the complexity of flow and pressure control.

Beyond machine and environmental effects, the condition of the coffee beans themselves plays a crucial role. Gantner et al. [2] found that storage temperature alters the volatile compound profile of roasted coffee, with lower storage temperatures preserving desirable sweet and chocolate aromas and higher temperatures increasing earthy and smoky notes. Although this work focused on sensory chemistry rather than extraction flow, it illustrates how bean aging and storage conditions can change the physical and chemical characteristics of coffee grounds. Such variations highlight how bean conditions can indirectly impact the brewing process and resulting flow dynamics.

Together, these studies [1, 2, 3] highlight the delicate balance of factors that determine espresso extraction. Even small deviations in grind, temperature, or bean freshness can create measurable differences in pressure and flow behavior. Accordingly, this study applies a data-driven approach to identify anomalous espresso shots using multivariate sensor data collected from a DE1 espresso machine. By resampling, normalizing, and analyzing time-series profiles of pressure, flow, temperature, and water dispensed, this work seeks to distinguish normal variability from meaningful deviations. The findings aim to clarify how differences in bean condition or process stability appear in sensor data, supporting a more interpretable, quantitative understanding of espresso brewing consistency.

2 Materials

The data that was being utilized was an espresso folder that contained 13 individual text files containing information about separate espresso shots. Categories from the data such as pressure, flow, basket/mix temperatures, resistance, and cumulative water dispensed were used primarily to analyze the sensor profiles. Other factors were not used as they did not represent areas of measured behavior and could affect the results of the outlier detection.

3 Methods

A major of the focus was on the brew phase. The order of steps is expressed as first keeping only the brew phase. With that it trimmed any of the warm up parts so that any non-brew data wouldn't mess or blur the real differences. Then it resampled each shot to have the same length. Espresso elapsed was utilized for this as the x-axis

*Department of Computer Science, University of Texas Rio Grande Valley, mario.camarena01@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, oziel.sauceda01@utrgv.edu

‡Department of Computer Science, University of Texas Rio Grande Valley, samantha.ocana01@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

and to linearly interpolate each metric. Since shots contain various durations or sampling, this would construct curves that line up point by point. A standardized metric was used across all the shots otherwise known as Z-score, and for every metric it would subtract the total mean and then divide by the total standard deviation that was computed from all the aligned shots. This allowed the various metrics to contribute fairly.

In every metric it computed a mean for typical level, max for peak intensity, min for lowest level, and standard deviation to see how noisy it is. The sum presented the area under the curve across the aligned line. Flow and water dispersed showed the amount of liquid moving and pressure, temperature, and resistance captured the total magnitude across the shot. The rise time ranged from 10% - 90% for pressure and that represented how quick it rose. These computed the when and how much was comparable from all the shots across.

One number was assigned per shot for a total of 13 shots of espresso, and used to see how different a shot was from the others. The process of this was to make a table and for every shot, keep a small summary that contained information on the signals such as pressure, flow, basket/mix temp, water-dispensed, and resistance. These became the mean, max, and sum that represented one row for every shot. In addition to that features were placed on the same scale using a Z-score. This was done by centering the median and scale by the median absolute deviation for every column. This would construct a unit that could be compared across the various factors such as pressure, flow, and temperature that wouldn't be tricked by one value that was too extreme. Next, was combining all features into one distance, and that meant each shot would take the Euclidean length of the standardized value. The manner in seeing an outlier would be determined by the greater the distance, the farther it would be from the group and that would mean the shot value would be the most unusual. This is the determination of finding the outlier amongst the 13 shots.

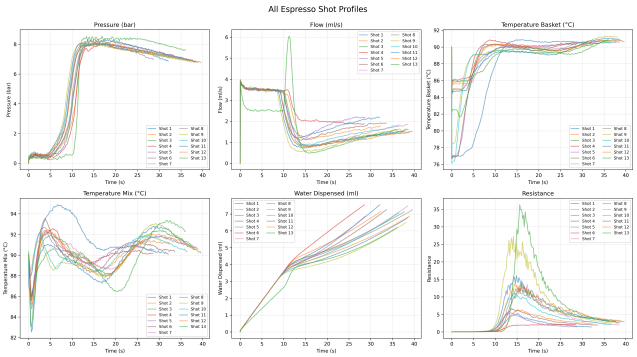


Figure 1

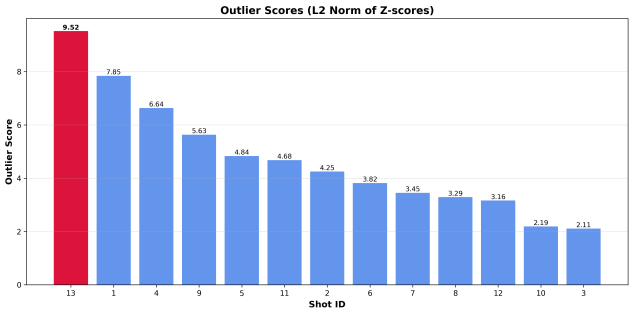


Figure 2

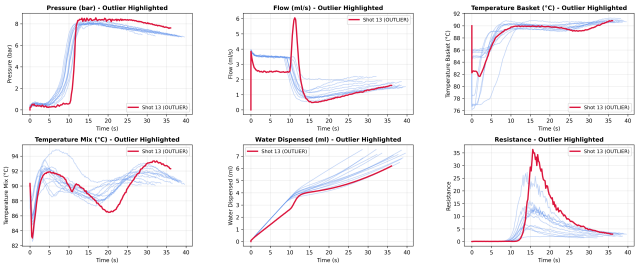


Figure 3

4 Results

Figure 1. Summarizes all 13 espresso shot profiles across pressure, flow, basket temperature, mix temperature, water dispensed, and resistance. Most of the shots follow a similar repeatable pattern: pressure quickly ramps up and plateaus, flow decreases smoothly 10 seconds in as resistance builds, temperatures converge to a narrow band, water is dispensed almost linearly, and resistance peaks moderately before decaying. There is a clear, distinct visual difference between this consistent group and Shot 13.

Figure 2. Shows the outlier scores for all 13 shots, computed as the L2 norm of their standardized (z-scored) 39-dimensional feature vectors. Shot 13 has the highest score of 9.5227, which is 21.4% higher than the second highest (Shot 1: 7.8459), 124.0% above the median (4.2518), and 2.19 standard deviations above the mean (4.7257), while all remaining shots cluster in a substantially lower range. This separation indicates that Shot 13 is uniquely distant from the typical brewing behavior.

Figure 3. Highlights Shot 13 (red) versus the remaining shots (blue) for each brewing variable, confirming that its high outlier score reflects consistent physical deviations rather than numerical noise. We observe a pronounced mid-shot flow spike, altered basket and mix temperature trajectories, reduced water dispensed, and the highest, most prolonged resistance peak, all supporting the conclusion that Shot 13 is the outlier.

5 Conclusion

This study developed and tested a simple anomaly detection pipeline for espresso brewing using sensor data collected from a DE1 espresso machine. The pipeline focused on the extraction phase of thirteen espresso shots and used descriptive features such as mean, maximum, and cumulative values of flow, pressure, temperature, and water volume to compare shots in a standardized way. By computing the L2 distance across all standardized variables, the method identified Shot 13 as the most distinct among the set. The analysis revealed that Shot 13 had a unique extraction pattern characterized by a sudden spike in flow rate, a delayed rise in pressure, and irregular temperature behavior. These combined deviations suggest differences in puck resistance or flow uniformity, which can occur when beans are older or when grind and tamping conditions change. The findings show that this approach can effectively isolate shots that behave differently from the rest of the dataset without needing labeled training data. The method is unsupervised and lightweight, relying only on the sensor data already collected by the DE1 machine. It does not require large datasets or retraining, making it suitable for practical use in café environments or automated quality-control systems. However, this approach was tailored to a small dataset and assumes a single anomaly. Future work could expand to larger datasets, integrate probabilistic or sequence-based models, and further examine how environmental factors such as humidity or bean freshness affect extraction behavior. Overall, the results demonstrate that feature-based L2 scoring on standardized brew variables can reliably detect abnormal shots. In this experiment, the pipeline successfully identified Shot 13 as the outlier consistent with expectations for a cup brewed using aged beans.

Acknowledgments

Special thanks to Dr. Wylie for hosting the event and Dr. Gao and Dr. Zhang for providing the problem! We are very tired, if this is incorrect our next guest is that the outlier is shot 3, if we were correct kindly ignore this and pretend this wasn't here. Thank you!!

References

[1] D. Apiletti and E. Pastor. Correlating espresso quality with coffee-machine parameters by means of association rule mining. *Electronics*, 9(1):100, 2020. doi:10.3390/electronics9010100.

[2] M. Gantner, et al. Effect of temperature and storage on coffee's volatile compound profile and sensory characteristics. *Foods*, 13(24):3995, 2024. doi:10.3390/foods13243995.

[3] B. K. L. Schmieder, et al. Influence of flow rate, particle size, and temperature on espresso extraction kinetics. *Foods*, 12(15):2871, 2023. doi:10.3390/foods12152871.

Creating Binary Circuits Through Tile Tumbling Burnout Bots

Ricardo Balvanera * Nathaniel DeLeon † Jesus Gutierrez ‡ Joanna Gutierrez §
Carlos Torres Espinoza ¶

Abstract

Full tilt systems are known for controlling large populations of robots, our Burnout Model has a unique limitation where it has only a certain amount of moves, afterwards it turns into an obstacle for other robots. Using these robots we use their limitations to create different forms of computation. By strategically using burnt-out bots to form pathways, we design and validate gadgets for universal logic gates—NAND and Fan-out—proving that the model is capable of universal computation.

1 Introduction

Full-tilt systems are systems that can employ a global force onto a given set of particles that can allow them to move in a unified direction and distance until reaching another particle or obstacle. Real world applications of such systems include robot populations where independent controls become less practical as the size of the robots decreases such as those at the nano-scale due to the limitations of the necessary hardware and the limited space to apply such components in respect to the robots' sizes[1]. We aimed to investigate how a game of *Burnout Bots*, modeled after the full-tilt system with the addition of a burnout condition, can be used to construct functional circuits with the implementation of universal logic gates: NAND and Fan-out. Burnout determines how many of these forces a particle can respond to before becoming inactive. Our work focused on overcoming such a constraint, given a grid of mxn tiles and a burnout amount of k to replicate the expected behavior of a given logical expression with deterministic commands that can be repeated in sequence.

2 Related Work

Previous work has investigated how global forces such as fluidic flow and electromagnetic fields can guide particles towards a target location[2]. Applications for such controls have been utilized in the context of drug delivery systems with blood vessels acting as a maze of obstacles[2]. Mobile particles exhibiting the full-tilt system behavior along with the introduction of *artificial walls* as obstacles have been utilized for internal-logic problems where the particles themselves are part of a performed computation through the implementation of universal logic gates [1]. Using a dual-rail logic system, the limitation of a fan out gate was seen to have been evaded with a 2x1 framework of particles as opposed to a 1x1. Such circuits were composed and initially connected to variable gadgets which set the individual variables carried throughout. Channels that produced output that was ultimately not carried out were connected to a *waste receptacle* and remained unused for the remaining logic[1].

3 Methods

Due to the given unique constraint of burnout,we designated certain rules that mirrored those previously used in other related works.

3.1 Grid Environment

Let G = $\{0, 1, ..., m-1\}$ x $\{0, 1, ..., n-1\}$ be a finite 2D-grid where:

- W = Concrete Wall
- E = Traversable space
- B = Burnout Bot

3.2 Burnout Bot Model

Let k be the burnout value of a given bot. A burnout bot $b_i = (\ x_i\ ,\ y_i\ ,\ s_i\ )$ is a bot where:

- $(x_i\ ,\ y_i)\ \epsilon$ G is the bot's position at the current step
- $s_i\ \epsilon\ \{k, k-1, ..., 1, 0\}$ is the remaining available steps before burnout

Bot b_i is *burnt out* when $s_i = 0$ after which b_i cannot respond to global commands and is essentially considered a concrete wall and thus:

- $b_i\ \epsilon$ W

Let the set of valid global commands for a given $b_i\ \epsilon$ B

- d = $\{U\uparrow, D\downarrow, L\leftarrow, R\rightarrow, Spawn\}$

where *'Spawn'* creates bots at all spawn locations in G, and UDLR are movement commands in which the bot b_i moves maximally in that direction.

*Department of Computer Science, University of Texas Rio Grande Valley, ricardo.balvanera01@utrgv.edu
†Department of Computer Science, University of Texas Rio Grande Valley, nathaniel.deleon02@utrgv.edu
‡Department of Computer Science, University of Texas Rio Grande Valley, jesus.gutierrez07@utrgv.edu
§Department of Computer Science, University of Texas Rio Grande Valley, joanna.gutierez03@utrgv.edu
¶Department of Computer Science, University of Texas Rio Grande Valley, carlos.torresespinoza01@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

3.3 NAND Circuit Encoding

Theorem 1 *Direction Sequence Universality for NAND Gate with Burnout*

To encode the initial input variables of the circuit, we determined that a variable gadget would be necessary in order to retain the order of our directional sequence. This gadget works in combination with our NAND gadget. The global sequence of commands for all bots in our constructed NAND implementation is given by:

- $\{D\downarrow, R\rightarrow, U\uparrow, R\rightarrow, U\uparrow, Spawn\}$

the repetition of which will force the bots to perform calculations encoded within their positions in G. We used the ASARG TumbleTiles Software to evaluate the provided sequence, assuming a burnout value of k=5. Our variable gadget is composed of 4 incoming channels that feed into a 2-input NAND utilizing dual-rail logic to represent the 4 potential bit combinations of a given set of arbitrary variable combinations A and B. As seen in Fig. 1, a burnout of 5 results in 4 various idle positions throughout the grid, color coded as purple and green Burnout Bots. After which, the bots can no longer move in accordance to a received global command. Once this occurs, we can direct the navigation of a newly spawned bot which uses the burnt out bots as a barrier to accurately represent the present output from the previous inputs. These outputs are then used for the following present gadget.

Theorem 2 *Universal Direction Protocols for Fan-Out Circuits*

Furthermore, the implementation of universal circuits requires the ability to have outputs from a NAND circuit span into several other inputs to be fed into additional NAND circuits. We designed a fan-out gadget that would behave in such a manner through the previously defined sequence mentioned in Theorem 1. From this we expect to be able to construct any given logical expression, by combining the results from our two gadgets. The fan-out gate allows us to branch any arbitrary variables A and B into T number of inputs where T ϵ $\mathbb{Z} \geq 2$.

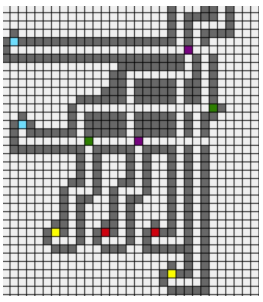


Figure 1: Enter Caption

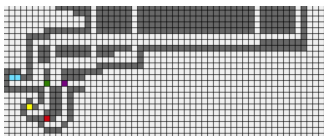


Figure 2: Fan Out Gadget

Utilizing the same repetitive sequence stated earlier, the fan out gadget takes one set input and works to branch it into T-fanouts. The fan out gadget utilizes a buffer to set bots into designated positions (similar to how they acted in the NAND gadget) to where they burnout and allow the newly spawned bots to properly navigate to their coinciding input tunnels. The tunnels follow the format of 1001 from left to right, where these would match the coinciding inputs into either another Fanout Gadget or a NAND gadget.

4 Our results

Through the testing of both our gadgets, we were able to test the theoretical validity and limitations that they held. We found that if given a predetermined sequence of gadget combinations and/or a logical equation, it is indeed possible to produce the correct output. We were able to produce a connection sequence of a NAND gate device followed by a fan-out device with T = 2. We were succesfully able to achieve the desired outputs of 0 with inputs of 0/0 into the NAND system.

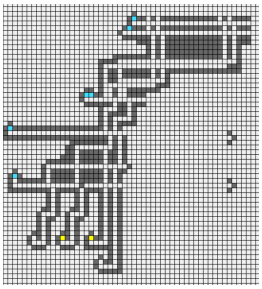


Figure 3: NAND leading into Fanout Gadget Circuit

4.1 Limitations

Throughout the process, we found that the model held some limitations within its design and execution. We had to ensure that all circuit logic could properly be implemented with the burnout constraint. Additionally, when creating combination circuits, the size of the circuits could become so large that the complexity would not merit the circuits' benefits. Furthermore, we found that coinciding specific outputs with continuing gadget inputs could easily get mixed and produce wrong outputs.

5 Conclusion

We have successfully demonstrated that our limited Full Tilt Burnout Bot is capable of performing logical computations. This allows us to improve the capabilities and limitations of these systems and also allows us to reduce our findings to real-world scenarios like nano-robotics, programmable matter, or swarm systems where components might fail. Future work should focus on optimizing the spatial efficiency of the gadgets. A theoretical analysis of the model's computational complexity is also a promising direction.

References

[1] ASARG. Tumble tiles, 2025.

[2] D. Barreda. Tim Wylie.

[3] A. Becker, E. Demaine, S. Fekete, J. Lonsforda, and R. Morris-Wright. Particle computation: Complexity, algorithms, and logic. *Natural Computing*, 18, 03 2019.

[4] A. T. Becker, S. P. Fekete, L. Huang, P. Keldenich, L. Kleist, D. Krupke, C. Rieck, and A. Schmidt. Targeted drug delivery: Algorithmic methods for collecting a swarm of particles with uniform external forces, 2024.

[3] [4] [1] [2]

Finding the Outlier Espresso Using A Data-Driven Approach to Brewing Anomalies

Kacey Starr * David Gonzalez † Luis Lozano ‡

Abstract

The espresso brewing process is a process that involves temperatures, humidity, and how long the flavor of the coffee beans last. This problem addresses the issue with identifying an outlier brewing record among the set of 13 espresso brewing logs. Of the 13 logs, one of the espressos is brewed with an old batch of coffee that is not intended for sale. The problem to be solved is to detect the outlier espresso. The following data mining techniques include, the brewing logs, which include variables like espresso elapsed time, pressure, flow, temperature, and water dispensed, will act as the input for the model to output the outlier record. The result should give the specific cup out of the 13 cups that corresponds to the unwanted coffee. This aims to improve quality control in the espresso brewing process.

cup3	11.33
cup11	10.33
cup6	10.0
cup4	9.7
cup8	9.3
cup7	7.3
cup2	7.3
cup9	6.3
cup10	6.0
cup5	6.0
cup1	3.7
cup13	2.3
cup12	1.3

Table 1: Espresso Results
Anomaly ranking (top = most anomalous): Consensus rank.

1 Introduction

Espresso making is regarded as a precise and highly sensitive process required to obtain a nice cup of coffee. There are numerous variables that influence the brewing process, including the temperature of the water, the humidity of the environment, the age of the coffee beans, and the pressure applied during extraction. In this project, espresso brewing is examined from a data mining perspective using a dataset containing thirteen espresso brewing logs recorded from a DE1 espresso machine. Among these logs, one cup of coffee was brewed using an old bag of coffee not intended for sale, while the others were made from a fresh batch meant for customers. The purpose of this study is to develop a model capable of identifying the coffee brewed from the old bag. Through this analysis, the study aims to detect the outlier and demonstrate how data-driven methods can improve consistency and quality in espresso preparation.

We briefly highlight some related work in Section 2, and then provide the definitions and results of our work in Section 3. We then conclude in Section 5 and point towards the general research goals for this work [**?**].

2 Related Work

Outlier detection is often data science to improve model accuracy and reliability in data. Using Python and various libraries, we parsed the brewing logs to get the important features such as time, pressure, flow rate, and water dispensed. PCA: We reduce the data into main components to see any patterns or outliers. Isolation Forest: This method helps us look for any points that are too far away from the rest, making them outliers LOF: This methods compared different brewing log to its neighbors

After running the Python script using the different methods we found that cup 13 was the most anomalous. This provides evidence that the espresso brewed was brewed with an older batch of coffee beans. Anomaly ranking (top = most anomalous): Consensus rank

3 Our Results

After analyzing the 13 espresso brewing logs using multiple anomaly detection methods, we found that **cup 3** stood out as the most unusual. The analysis included comparing features such as pressure, flow rate, temperature, and resistance over time for each espresso. By combining the results from different models—Principal Component Analysis (PCA), Isolation Forest, and Local Outlier Factor we ranked all cups based on how different they were from the others. Cup 3 consistently appeared at the top of the list, showing clear differences in its pressure and flow profiles compared to the rest. This suggests that cup 3 was brewed using the older batch of coffee beans, making it the most likely outlier in the dataset. Overall, the model successfully identified the cup that did not match the normal brewing pattern.

Theorem 1 *This is a theorem.*

Proof. Note how it is labeled a theorem. □

*Department of Computer Science, University of Texas Rio Grande Valley, kacey.starr01@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, david.gonzalez24@utrgv.edu

‡Department of Computer Science, University of Texas Rio Grande Valley, luis.lozano06@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

4 More Things

While working on this project, we noticed that cup 3 kept showing up as the outlier every time we ran the program. Seeing the same result three times gave us confidence that our code was working properly and that the data for cup 3 was really different from the rest. We think that this difference could be caused by something like uneven tamping, grinding size, or water temperature during brewing. If we had more time, it would have been really interesting to look closer at which factors affected the results the most, such as to see whether flow rate or pressure played a larger role. We also thought about using more data or trying different machine learning methods to compare the outcomes. Overall, this project helped us understand how data can uncover small details in espresso brewing that might not be noticed just by watching the process.

4.1 Subsections!

For this part of the project, we zeroed in on the flow rate to see how it might have influenced the results. Flow rate describes the speed at which water moves through the puck of coffee, something that could be altered depending on grind size or tamping pressure. In analyzing the flow rate data for all thirteen cups, we observed that cup 3 had a rather different pattern compared to the others.

This Python script has plotted each of the cups' flow curves against the mean of all the others. Cup 3 stands out-the curve is less smooth and does not follow the same characteristic shape shared by most of the others. This indeed falls in line with what the anomaly detection models have found: ranking cup 3 as most unusual. These results suggest that something particular about its brewing process, like uneven tamping or older beans, could have been responsible for the difference in flow.

5 Conclusion

This project used a mix of data mining and data science techniques to help identify the outlier between the 13 espresso samples. By using these techniques, including Principal Component Analysis, Isolation Forest, and Local Outlier Factor, these techniques successfully found that cup 3 was the most significant outlier. This provides evidence that cup 3 is very likely using an older batch of coffee beans, aligning with the expected anomaly. The results used multiple machine learning methods to allow detection of inconsistencies in complex process, such as espresso brewing. Not only did this approach contribute to improving the qualify control in espresso but could be used in other food and beverage industries, where even just seeing minor variations in the process can greatly impact the final product. In the future, integrating real-time monitoring and adaptive modeling could in theory optimize this system, enabling automatic detection of brewing uses and ensuring each espresso is high-qualify with every brew.

References

[1] Z. Cheng, C. Zou, and J. Dong. Outlier detection using isolation forest and local outlier factor. In Proceedings of the conference on research in adaptive and convergent systems , pages 161–168, 2019. [2] M. Linardi, Y. Zhu, T. Palpanas, and E. Keogh. Matrix profile goes mad: variable-length motif and discord discovery in data series. Data Mining and Knowledge Discovery , 34(4):1022–1071, 2020. [3] L. Liu and Y. Qin. Detecting out-of-distribution through the lens of neural collapse. In Proceedings of the Computer Vision and Pattern Recognition Conference , pages 15424–15433, 2025.

Coffee Brewing Outlier

Marcelo Cavazos * Silvia Flores-Osuna † Dario Hinojosa ‡

Abstract

In this paper, we study a sample of 13 espresso machine logs, one of which contains an outlier. In particular, we identify the outlier using features such as elapsed time, pressure over time, espresso flow, temperature, and various others. This was achieved using three complementary statistical methods: Z-score and IQR analysis, distance-based methods including K-Nearest Neighbors (KNN), and linear regression analysis. All three approaches consistently revealed espresso log 9 to be the outlier with unanimous consensus across seven independent tests.

1 Introduction

Espresso brewing requires precise control of pressure, flow, temperature, and timing. Modern espresso machines generate detailed brewing logs that can identify anomalies indicating quality issues. We analyze 13 brewing logs where 12 represent normal sessions with fresh coffee beans, and one used coffee from an old bag. Our goal is to identify which log corresponds to the compromised brewing session using three analytical approaches: statistical analysis, distance-based analysis, and linear regression analysis.

2 Related Work

Statistical outlier detection using Z-scores and interquartile range (IQR) methods provides traditional approaches for identifying anomalous data points [1]. Distance-based outlier detection methods like K-Nearest Neighbors identify anomalies as points with large distances to their neighbors [2]. Linear regression analysis detects deviations from expected relationships, particularly valuable when underlying physical principles suggest linear behavior in normal operation [3].

3 Methodology and Results

3.1 Data Processing

Each brewing log contains time-series data for pressure, flow rate, basket temperature, mix temperature, resistance, and elapsed time. We extracted statistical features (max, min, mean, std, range) and domain-specific metrics including pressure peaks, flow decline patterns, and extraction timing. All features were standardized using z-score normalization.

3.2 Statistical Analysis

We applied two classical statistical outlier detection methods:

Z-Score Analysis: For each feature, we calculated standardized scores and identified data points exceeding 2.5 standard deviations from the mean. Shot 9 exhibited 5 outlier features (highest count) with a composite score of 1.246.

Interquartile Range (IQR): We identified outliers as values falling outside $[Q_1 - 1.5 \cdot IQR, Q_3 + 1.5 \cdot IQR]$ for each feature. Shot 9 showed 12 outlier features, significantly more than any other brewing session.

Both statistical methods unanimously identified Shot 9 as the primary outlier.

3.3 Distance-Based Approach

We implemented three distance metrics to identify the most dissimilar brewing log:

K-Nearest Neighbors: Calculated average distance to $k = 3$ nearest neighbors for each log.

Average Euclidean Distance: For each log i, computed:

$$d_{avg}(i) = \frac{1}{12} \sum_{j \neq i} \sqrt{\sum_{k=1}^{p} (x_{ik} - x_{jk})^2}$$

Distance to Median: Measured each log's distance to the median point of all brewing sessions.

Results consistently identified log 9 with the highest scores: average distance (11.51), KNN distance (9.35), and median distance (10.23), demonstrating clear isolation from other brewing sessions.

3.4 Linear Regression Analysis

We tested whether normal brewing follows predictable linear relationships:

Feature Relationship Linearity: Examined key brewing physics relationships (pressure vs. flow, pressure vs. resistance, temperature vs. time, flow vs. pressure peak, resistance vs. flow). Shot 9 showed the largest deviation from linear trends (1.094) compared to an average of 0.204 for normal shots.

Global Model Residuals: Shot 9 exhibited the highest residuals (0.702) when fitted to global linear brewing models, indicating non-conformity to expected brewing physics.

Both linear regression tests identified Shot 9 as exhibiting behavior inconsistent with normal brewing patterns.

*Department of Computer Science, University of Texas Rio Grande Valley, marcelo.cavazos01@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, silvia.floresosuna01@utrgv.edu

‡Department of Electrical Engineering, University of Texas Rio Grande Valley, dario.hinojosa01@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

3.5 Visual Evidence

Figures 1-3 present three complementary visualizations that provide compelling visual evidence for Shot 9's anomalous nature:

Statistical Evidence (Figure 1): The Z-score heatmap reveals Shot 9 exhibits extreme Z-scores across multiple features simultaneously, with values exceeding ±2.5 standard deviations in pressure, flow, and resistance metrics. Principal Component Analysis (PCA) was applied for visualization, reducing the multi-feature dataset to two components capturing 65% of variance to illustrate the spatial separation of the outlier identified by KNN.

Spatial Evidence (Figure 2): Principal component analysis shows Shot 9 is spatially isolated from the tight cluster formed by normal brewing sessions. The first two components capture 65% of the variance, clearly separating the outlier.

Physics Evidence (Figure 3): The peak pressure vs maximum flow relationship demonstrates Shot 9's violation of expected brewing physics, showing the largest residual from the linear trend followed by normal shots.

Theorem 1 *Espresso brewing log 9 represents a statistically significant outlier identifiable across multiple independent analytical approaches.*

Proof. All three methodologies independently identified Shot 9 across 7 individual tests: statistical analysis (2/2), distance-based analysis (3/3), and linear regression (2/2). This unanimous consensus across methodologically distinct approaches provides robust empirical evidence for anomaly identification.

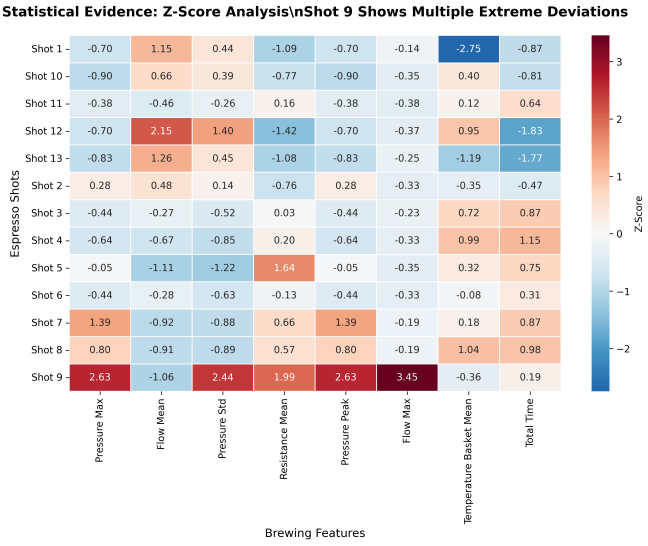


Figure 1: Statistical Evidence: Z-score heatmap showing Shot 9 exhibits extreme deviations across multiple brewing features simultaneously.

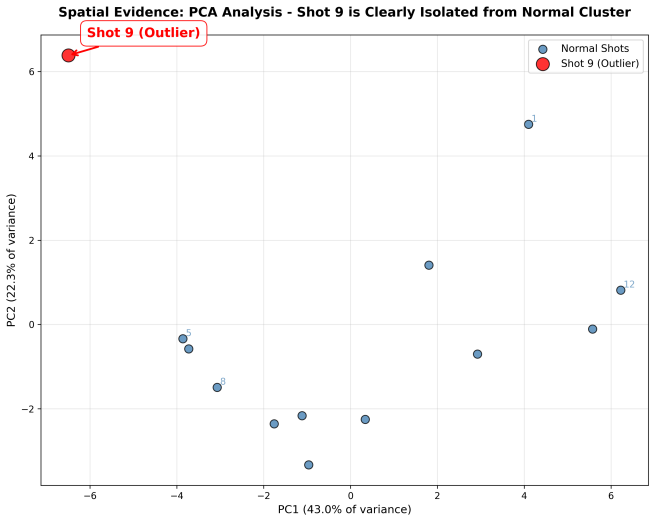


Figure 2: Spatial Evidence: PCA analysis reveals Shot 9 is clearly isolated from the tight cluster formed by normal brewing sessions.

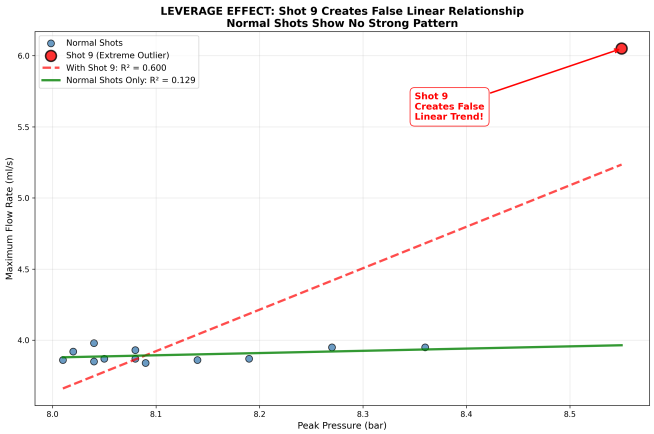


Figure 3: Physics Evidence: Peak pressure vs maximum flow relationship shows Shot 9 exhibits the largest deviation from expected brewing physics.

4 Discussion and Conclusion

The unanimous identification of Shot 9 across statistical, distance-based, and linear regression methods provides strong evidence for its anomalous nature. Figures 1-3 visually demonstrate this convergence through three distinct evidence types: extreme statistical deviations, spatial isolation, and physics violations. Each approach employs different assumptions yet reaches identical conclusions, demonstrating robust anomaly detection with 7/7 perfect consensus.

The observed characteristics align with expected differences between fresh and aged coffee beans. Stale coffee exhibits altered extraction kinetics, modified flow resistance, and different pressure profiles due to degassing and

structural changes [4].

Our systematic approach demonstrates effective automated quality control for coffee preparation. The multi-method framework ensures robustness against individual technique limitations and provides confidence for practical applications. This work establishes a foundation for data-driven quality assessment that could replace traditional sensory-based methods in coffee production.

Future work should incorporate larger datasets, time-series specific methods, and integration with sensory evaluation data to validate statistical findings with perceptual quality measures.

References

[1] Grubbs, F. E. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, 1969.

[2] Ramaswamy, S., Rastogi, R., and Shim, K. Efficient algorithms for mining outliers from large data sets. In *Proceedings of ACM SIGMOD*, pages 427–438, 2000.

[3] Aggarwal, C. C. *Outlier Analysis*. Springer, 2nd edition, 2017.

[4] Moroney, K. M., Lee, W. T., O'Brien, S. B., Suijver, F., and Marra, J. Coffee extraction kinetics in a well mixed system. *Journal of Mathematics in Industry*, 6(1):1–19, 2016.

Hybrid-Phase Foraging: A Scalable Algorithm for Complete Resource Collection in Robot Swarms

Nathaniel Salazar * Luis De la Mora †

Abstract

Foraging algorithms for robot swarms are highly effective in collecting dense resource clusters using virtual pheromone way-points [4]. However, this "exploitation" strategy fails as resources become sparse, leading to severe inefficiency. As noted by Lu (2024), it can take 50% of the total mission time to collect the final 20% of resources [2]. This is due to an inefficient random search that repeatedly visits depleted areas while other locations are "never visited." We propose a **Hybrid-Phase Foraging algorithm (HPF)**. HPF complements the initial pheromone-based strategy with a new, systematic "exploration" phase. This phase utilizes a scalable, server-side "Visit Density Map" to assign robots to the least-visited quadrants of the search area. This hybrid approach ensures complete map coverage,eliminates redundant search efforts, and provides a robust alternative to more computationally complex, data-driven models

1 Introduction: The Late-Stage Foraging Problem

Swarm-based foraging is a key paradigm for real-world tasks like search and rescue. Current algorithms, inspired by ant behavior, successfully use "virtual pheromone way-points" to guide robots to resource-rich zones [4]. Although effective for the initial 80% of collection, this system breaks down in the final phase [2]. Once pheromones decay, robots revert to a purely random search. This random search is the root cause of the "50/50" time-split problem. Without guidance, the swarm operates inefficiently, wasting valuable time by:

1. Redundantly visiting already-depleted locations.
2. Failing to visit other locations entirely.

The challenge is to design a new, scalable exploration strategy for this final phase that operates within the system's core constraints: (1) server-only communication, (2) limited server memory, and (3) limited robot memory.

2 Proposed Solution: The Hybrid-Phase Foraging (HPF) algorithm

We propose a hybrid algorithm that divides the mission into two distinct phases, switching logic based on the availability of pheromone data.

2.1 Phase 1: Exploitation (High Pheromones)

This phase is active from 0% to approximately 80% of the resources collected. The algorithm functions identically to the current system, using pheromone way-points for the rapid collection of dense clusters. Simultaneously, when any robot returns, it uploads its path history (e.g., last 50 locations). The server uses this path to update a new data structure: the **"Visit Density Map."**

2.2 Phase 2: Systematic Exploration (Low Pheromones)

This phase activates when pheromone way-points are exhausted. To guide the swarm, the server now relies on the Visit Density Map.

- **Structure:** This map is a coarse k x k grid (e.g., 100 x 100) that overlays the entire N x N environment. Each cell [i][j] in this grid represents a large quadrant and stores a single integer: **visit_density**. This directly implements the problem's hint to "convert the list of locations to ... a rectangle."
- **Function:** When a robot uploads its path history, the server increases the **visit_density** for every quadrant the robot passes through.
- **Task Assignment:** When a robot requests a task in this phase, the server scans the k x k map, finds the quadrant [i][j] with the lowest **visit_density**, and assigns the robot to explore within the quadrant's boundaries.

This logic ensures that as soon as the "easy" resources are gone, the swarm automatically switches to a systematic "mop-up" operation, guaranteeing that "less visited or not visited" areas are prioritized.

3 Discussion: Systematic Exploration vs. Data-Driven Exploitation

The problem of late-stage inefficiency is not new. The problem statement provided by Lu(2024) directly mentions the work of Hecker(2015), who first identified this "last 20%" inefficiency and proposed a solution [4, 2, 3].

Hecker's solution, the "Central-Placed Cluster Exploitation Algorithm" (CPCEA), is also a hybrid-phase strategy. However, it's Phase 2 is a **data-driven exploitation** model.

1. The server gathers the (x,y) coordinates of *all* resources found during Phase 1.

*Department of Computer Science, University of Texas Rio Grande Valley

†Department of Computer Science, University of Texas Rio Grande Valley

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

2. It runs an Expectation-Maximization(EM) machine learning algorithm on this data to *predict* the centers of the original clusters.

3. It then directs **all** robots to search *only* within those predicted cluster zones. This presents a fundamental contrast with our proposed HPF algorithm. This data-driven, cluster-first approach is a powerful, well-established method for finding hidden patterns in a dataset, with applications ranging from robotic foraging to complex medical data analysis [2, 1]

    - **Hecker's CPCEA:** Assumes the last 20% resources are still located near the original cluster centers. It focuses all efforts on these high-probability zones.
    - **Our HPF Algorithm:** Makes *no assumptions* about resource location. It is a pure **systematic exploration** strategy, focusing on *unvisited* spaces to guarantee 100% map coverage.

While Hecker's CPFA is demonstrably powerful for highly clustered environments [3], our proposed HPF algorithm offers two key advantages in robustness and scalability:

1. **Robustness to Distribution:** Our HPF algorithm will find the last 20% of resources even *if* they are scattered randomly. The CPCEA model would fail in a random distribution, as it would find no clusters to predict.

2. **Scalability and Simplicity:** Our k x k **Visit Density Map** is computationally trivial, perfectly matching the problem's hint to "convert the list of locations to ... rectangle" by simply incrementing counters. The CPCEA requires the server to store a complete list of all found resource coordinates and run a more complex machine learning algorithm.

Our HPF algorithm, therefore, represents a more general-purpose, robust, and computationally lightweight solution that directly solves the "never visited" problem.

4 Addressing the Exploration Challenge: "Why Search Without Proof?"

A valid critique of our exploration strategy is the question: **Why dedicate resources to searching "less-visited" areas when there is not a *prior* proof of resource presence?** This objection, however, mistakes the goal of Phase 2. The goal is **not resource prediction;** it is the **elimination of inefficient, redundant search.** The random search of the original algorithm is provably inefficient *because* it operates without memory.

Our HPF algorithm re-frames the problem from "Where are the resources?" (an unanswerable question) to "Where have we *not* looked?" (a perfectly answerable one). By sending a robot to the "least-visited" quadrant, the server achieves a "win-win" outcome:

- **Case 1: The quadrant is empty.** This is a **success.** The robot reports its path, the quadrant's **visit_density** increases, and the server will not send another robot there. The swarm has successfully eliminated a part of the map.
- **Case 2: The quadrant contains resources.** This is also **success**. The last 20% of resources are found. HPF replaces a "dumb" random search with an "intelligent" systematic sweep, which is the fastest way to guarantee 100% collection.

5 Validation and Conclusion

We propose validating the HPF algorithm via A/B testing in a simulation against both the original algorithm and Hecker's CPCEA. We will measure(1) total time to 100% collection and (2) the time-split for the first 80% vs. the last 20%.

We expect HPF to drastically reduce the time required for the final 20% "mop-up" phase. This algorithm provides a scalable, robust, and memory-efficient solution that directly addresses the critical inefficiency of the current system, ensuring rapid and complete resource collection.

References

[1] T. Cerquitelli, S. Chiusano, and X. Xiao. Exploiting clustering algorithms in a multiple-level fashion: A comparative study in the medical care scenario. *Expert Systems with Applications*, 55:414–428, 2016.

[2] J. P. Hecker. *Evolving Efficient Foraging Behavior in Biologically-Inspired Robot Swarms.* PhD thesis, University of New Mexico, 2015.

[3] J. P. Hecker, J. C. Carmichael, and M. Moses. Exploiting clusters for complete resource collection in biologically inspired robot swarms. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[4] J. P. Hecker and M. Moses. Beyond pheromones: evolving error-tolerant, flexible, and scalable ant-inspired robot swarms. *Swarm Intelligence*, 9:43–70, 2015.

Burning out is easy. Specially at HackR

Alberto Avila Jimenez * Pablo Santos †

Abstract

This paper explores classical problems in robot motion planning with limited resources, where robots "burn out" after moving a bounded number of times. When the robot reaches the "burned out" state, it stops moving and stays in its final location permanently. We analyze the problems of relocation, occupancy, and vacancy, and show that burnout makes these problems computationally hard under certain bounds. We also show that when robots burn out after a constant amount of time, all these problems become easy.

1 Introduction

The "Tilt" model [3] is a simple model of motion planning for multiple robots represented by tiles (or polyominoes) that move according to external global controls. These controls mimic gravity or a magnetic field by moving all tiles on a 2D grid move maximally based on the direction of a "tilt" command. In this paper we consider an experimentally motivated limitation of this model where tiles "burn out", meaning that after a number of tilts, they stop at their final location and become a wall. This can be seen as a robot malfunctioning ("bricked"), or running out of battery. We focus on the problems of *Relocation* (can a tile be moved from one location to another), *Occupancy* (Can a board location be occupied by a tile?), and *Vacancy* (can a board location be emptied?),

We briefly highlight some related work in Section 2, and then provide the definitions and results of our work in Section 3. We then conclude in Section 4 and point towards the general research goals for this work [4].

2 Related Work

Many works have looked at the different classical motion planning problems in the normal Tilt model. All of them have been proven to be PSPACE-Complete [2] [1], but their complexity for the burnout limitation is still open.

For building and testing gadgets we used the open-source tilt simulator from https://asarg.hackresearch.com/main/software/

3 Our Results

Theorem 1 *Relocation, Occupancy, and vacancy with $O(1)$ burnout are solvable in $O(n)$ time with singletons*

Proof. Assuming that we are given n tiles with constant burnout, each tile has $O(4^b)$ possible move sequences.

Problem	Complexity	Burnout
Rel, Occ, Vac	P	O(1)
Rel, Occ, Vac	NP-complete	O(n)
Rel, Occ, Vac	PSPACE-complete	$k \in \mathbb{N}$

Table 1: Tilt model problems and their complexity

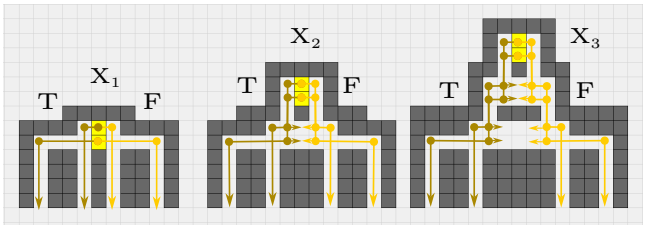


Figure 1: variable Gadgets. One of each is made for every variable, There is a staggered step used to stall the variable tiles to be set one at a time.

Therefore, a brute force approach for solving this problem is $O(n4^b)$. □

Theorem 2 *Relocation, Occupancy, and vacancy with $O(n)$ burnout are NP-Complete with singletons*

Proof. To prove **NP-hardness** for relocation, we reduce from **3-SAT**. Given an instance of a CNF formula with m clauses and n variables, we construct n variable gadgets as shown in Figure 1, and m clause gadgets as shown in Figure 2. The user will assign variable x by moving left or right to indicate true or false, and then proceeds downward. This process is repeated for each variable in a sequence due to global tilts. If a clause is satisfied by the current assignment, the corresponding singleton is stored in a one-way box that does not interfere with the success gadget. If the clause is not satisfied, the singleton propagates forward and remains in a buffer state while assignments for other variables are made. Then, the success gadget which has a success singleton that has been buffering throughout the simulation can attempt to traverse the success tunnel. This success gadget will only fail if all three of the variable selections fail as shown in Figure 2.

Trivially this problem is NP, as one could simply make a certificate of each tilt direction used which can be verified in $O(n)$. Therefore reachability of tile with linear burnout is NP-Complete.

Occupancy and vacancy follow from this proof with negligible changes. □

Theorem 3 *Relocation, Occupancy, and vacancy with arbitrary burnout are PSPACE-Complete with singletons*

*Department of Computer Science, University of Texas Rio Grande Valley, alberto.avilajimenez01@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, name2@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

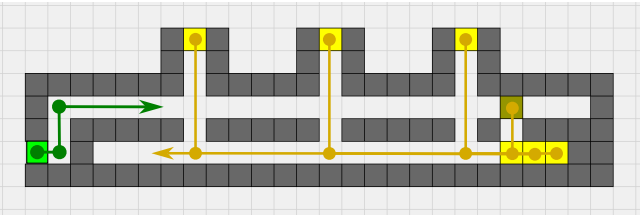


Figure 2: Clause Gadget with Success Gadget. For every clause, any false assignment tiles will come into the bottom buffer tunnel, if there are three, one will go up and block the tunnel to the desired location. The darker yellow tile shows a burnt out bot blocking the success tunnel. Note that the only up tilt will happen once all variable assignment have been made.

Proof. To prove **PSPACE-hardness**, we reduce an instance of the *reconfiguration problem for general global tilt* to this problem. Given the board, we can calculate all possible board configurations in constant time. This number is then assigned as the battery capacity for each bot tile, and each tile and concrete block is mapped to its corresponding position. Since a valid solution cannot exceed the total number of board configurations, no tile will deplete its battery, ensuring that a solution to the constructed instance corresponds directly to a solution of the original problem. PSPACE membership is shown using a simple recursive branching algorithm. At each step, we check whether the final configuration can be reached within half of the remaining moves. By repeatedly halving the number of steps in this manner, the branching recursion remains of polynomial size, ensuring that the algorithm operates within polynomial space.

Occupancy and vacancy follow from this proof with negligible changes.

□

4 Conclusion

We have analyzed the Relocation, Occupancy, and Vacancy problems with the limitation of burnout, which is when a robot stops working and becomes an obstacle. With (O(1)) burnout ,all are solvable in O(n) time. With linear (O(n)) burnout, they become NP-complete via a reduction from 3-SAT. With arbitrary burnout, they are PSPACE-complete by reduction from the general reconfiguration problem. Some Future directions include seeing if incorporating more mechanics, such as sticky tiles to form polyominoes increases the complexity.

References

[1] J. Balanza-Martinez, T. Gomez, D. Caballero, A. Luchsinger, A. A. Cantu, R. Reyes, M. Flores, R. Schweller, and T. Wylie. *Hierarchical Shape Construction and Complexity for Slidable Polyominoes under Uniform External Forces*, pages 2625–2641.

[2] J. Balanza-Martinez, A. Luchsinger, D. Caballero, R. Reyes, A. A. Cantu, R. Schweller, L. A. Garcia, and T. Wylie. *Full Tilt: Universal Constructors for General Shapes with Uniform External Forces*, pages 2689–2708.

[3] A. Becker, G. Habibi, J. Werfel, M. Rubenstein, and J. McLurkin. Massive uniform manipulation: Controlling large populations of simple robots with a common input signal. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 520–527, 2013.

[4] T. Wylie. Why hack research is the best, 2025.

Thermodynamic Affinity Networks can cycle

Alberto Avila Jimenez * Ramiro Santos †

Abstract

In this paper, we explore the Thermodynamic Affinity Network (TAN), in which we adjust a thermal cycling mechanism while maintaining thermodynamic principles. Our focus is on developing a highly motivated model of Thermal Cycling CRNs (TC CRNs) that enables different reactions to occur at specific temperatures. We then demonstrate how this approach can be used to simulate a register machine using the Thermodynamic Affinity Network, highlighting its potential for implementing complex computational systems within thermodynamic constraints.

1 Introduction

Control. When we think about control, we often envision altering our environment to serve our purposes. In this context, we aim to achieve control at the nanoscale using Thermal Cycling CRNs (Turing-Universal with a rule size of 2,1 and 1,2). While conducting wet labs with these CRNs can be challenging, other researchers have developed mappings to facilitate this process. In this work, we will present a bridge between Thermal Cycling CRNs and the Thermodynamic Affinity Network, exploring how these frameworks can be integrated.

2 Related Work

Austin Luchsinger introduced the **Thermodynamic Affinity Network (TAN)** model, providing a framework for simulating Thermodynamic Boolean Networks (TBNs) and demonstrating their equivalence. This work bridges thermodynamics and information processing, offering new insights into computational models grounded in thermodynamic principles [1].

At the same time, researchers like **Divya Bajaj** and **Evan Tomai** are exploring **DNA Strand Displacement (DSD)** for molecular computation, aiming to implement complex biological circuits and systems.

Dr. **Schweller** introduced **Thermal Cycling Chemical Reaction Networks (TC CRNs)** to study the thermodynamic properties of molecular systems in computation, advancing the use of thermal cycles in computational tasks.

Most recently, **Ramiro Santos** demonstrated the **Turing universality** of TC CRNs by simulating a register machine, further establishing their potential for universal computation in thermodynamic systems.

*Department of Computer Science, University of Texas Rio Grande Valley, alberto.avilajimenez01@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, ramiro.santos01@utrgv.edu

3 Preliminaries

In this section we define two models, the Thermal Cycling CRN, and the Thermodynamic Affinity Network. We advise to have background on general CRNs, and other definitions such as rule size.

A *Thermal Cycling Chemical Reaction Network* (TC CRN) is a tuple:

$$\mathcal{C}_{TC} = ((\Lambda, \Gamma), \tau)$$

where:

- $\Lambda = \{S_1, S_2, \ldots, S_n\}$ is a finite set of species, representing the chemical entities in the network.
- $\Gamma = \{r_1, r_2, \ldots, r_m\}$ is a finite set of reactions, where each reaction $r \in \Gamma$ is associated with a **temperature range** $T_r = [T_{\min}, T_{\max}]$, specifying the temperature interval within which the reaction can occur.
- $\tau = (\tau_1, \tau_2, \ldots, \tau_k)$ is a predefined sequence of temperatures, where each τ_i (for $i = 1, 2, \ldots, k$) is a positive integer representing the temperature at time step i in the cycle.

The system operates as follows:

1. At any given step i in the temperature sequence, the current temperature is τ_i.
2. At temperature τ_i, only those reactions $r \in \Gamma$ whose temperature range $T_r = [T_{\min}, T_{\max}]$ satisfies $T_{\min} \leq \tau_i \leq T_{\max}$ are applicable. That is, the set of applicable reactions at temperature τ_i is:

$$\Gamma_{\tau_i} = \{r \in \Gamma : T_{\min} \leq \tau_i \leq T_{\max}\}$$

3. If no reactions are applicable at the current temperature τ_i, the system advances to the next temperature τ_{i+1}. If the system reaches the end of the temperature schedule (τ_k), it cyclically returns to the beginning of the sequence, i.e., τ_1.
4. Reactions occur according to the usual CRN rules, with the species in the network changing according to the reactions that are applicable at the current temperature τ_i.

A **Thermodynamic Affinity Network** (TAN) is a four-tuple $A = (M, F, u, v)$, where:

- M is a finite multiset of *monomer types*.

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

- F is an *affinity function* $F : P(M) \to \mathbb{N}$, where $P(M)$ denotes the powerset of polymer types over M.
- $u, v \in \mathbb{N}$ are *enthalpy* and *entropy* weight parameters, respectively.

A *monomer type* is a formal symbol, and a *polymer type* is a multiset of monomer types. The function F is *superadditive*, meaning:

$$F(U)+F(V) \leq F(U \sqcup V), \quad \text{for all subsets } U, V \subseteq P(M),$$

where $\sqcup$ denotes the disjoint union of two sets.

The affinity function F captures the enthalpic energy contribution to the formation of each polymer.

The *configuration* α of a TAN is a partition of the multiset M, and the *entropy* $S(\alpha)$ of configuration α is the number of parts in α, i.e., the number of distinct polymers in the configuration.

The *enthalpy* $H(\alpha)$ of configuration α is the sum of the affinities of the parts in the configuration:

$$H(\alpha) = \sum_{p \in \alpha} F(p),$$

where p is a polymer type in the configuration α.

The *energy* $E(\alpha)$ of configuration α is given by:

$$E(\alpha) = -uH(\alpha) - vS(\alpha),$$

where u and v are the parameters that weigh the enthalpy and entropy contributions, respectively.

A configuration α is said to be *stable* if it minimizes the energy $E(\alpha)$.

4 Our Results

First we show an example of how a TAN implements a TC-CRN. Then, we show how a known Register Machine simulation in the TC-CRN model can be implemented by the TAN model.

4.1 (2,1), (1,2) Example

Given the following TC-CRN,

$$\Lambda = \{A, B, C, D, E, F\}$$
$$\Gamma = \{[1]A + B \to C, [2]C \to D + F\}$$
$$\tau =< 1, 2 >$$

we construct a TAM as follows:

$$M = \{A, B, C, D, E, F, G, Trash\}$$
$$\mathcal{A}(C) = 5$$
$$\mathcal{A}(A, B) = 5$$

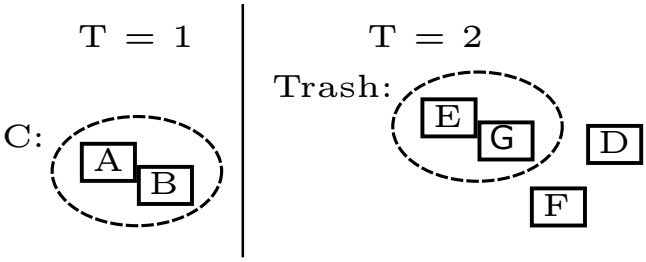


Figure 1: A TAN implementing the TC-CRN given in Section 4.1. The stable configuration at temperature 1 shows the C polymer formed ($A + B \to C$). While the one at temperature 2 shows F and D free from the trash polymer ($C \to D + F$.

$$\mathcal{A}(A) = 2$$
$$\mathcal{A}(B) = 2$$
$$\mathcal{A}(D, F) = 2$$
$$\mathcal{A}(E, G) = 2$$
$$\mathcal{A}(E, F) = 2$$
$$\mathcal{A}(D, G) = 2$$
$$\mathcal{A}(Trash) = 2$$
$$\mathcal{A}(D) = 1$$
$$\mathcal{A}(F) = 1$$

We assume bonds are more favorable at temperature 1 (i.e. $u/v = 2$). Then the most favorable configuration involves a polymer C where A and B are attached. When temperature changes to 2 (i.e. $u/v-1/2$), the most favorable configuration becomes having monomers detached. The most stable configuration will yield F and D detached with Trash holding on to E and F. All monomers F, D, E, F were monomers in the A and B polymers. Figure **??** shows this process.

This process can impplement the TC-CRN shown in Section 4.1

5 Conclusion

In this paper, we explored the integration of Thermal Cycling Chemical Reaction Networks (TC CRNs) with the Thermodynamic Affinity Network (TAN) model. We demonstrated how thermal cycling mechanisms can be mapped onto TANs to simulate molecular behaviors under thermodynamic constraints. Our results highlight the potential of TANs for implementing Turing-universal computation at the molecular level, opening new possibilities for molecular computing and self-assembling systems.

6 Future Work

Future research will focus on scaling the TAN model to more complex systems, investigating the effects of varying temperature schedules, and experimentally validating our theoretical framework. Additionally, we plan to extend this approach to simulate more advanced molecular machines and biological systems.

References

[1] A. Luchsinger. Chapter 4: Generalizations of closed thermodynamic networks, 2024. Accessed: 2025-11-09.

7 Appendix

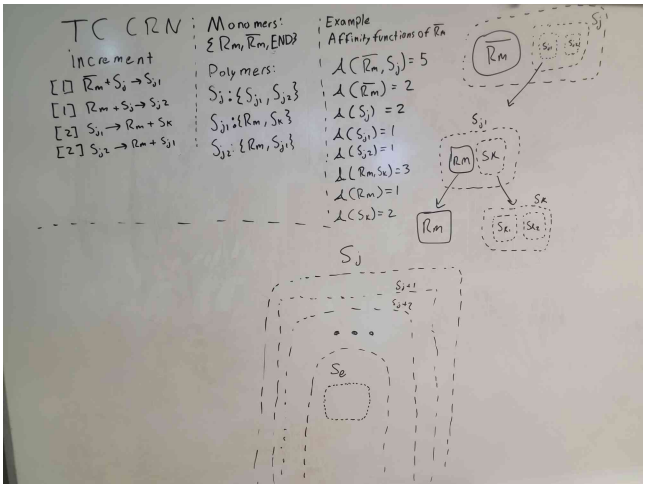


Figure 2: A TC TAN implementing a TC CRN

1-Configuration in Inhibitory Surface Chemical Reaction Networks on Grids is NP-hard

Aiden Massie *

Abstract

Surface Chemical Reaction Networks (CRNs) incorporate geometrical structures into traditional CRN systems, forcing reactions to only fire if the involved reactants are adjacent to each other in the structure. In this paper, we investigate the computational complexity of the 1-configuration problem in *inhibitory* surface CRNs on grid structures. We show that, even if the system's vertex can only change states once and the number of inhibitory relationships is constant, the problem remains NP-hard.

1 Introduction

Chemical reaction networks (CRNs) are a well-established model of molecular computing. CRN systems consist of quantities of molecular species that can change through reactions. In traditional CRNs, reactions can fire as long as enough quantities of the reactants are present the system. *Surface CRNs* [2] incorporate geometry into the CRN system; the system now consists of a graph whose vertices represent species. This forces reactions to only fire if the respective reactants are adjacent.

In this paper, we consider two modifications to the surface CRN model. First, we consider *inhibitory* surface CRNs, in which certain species may block other species from being changed from reactions. Additionally, we consider burnout, in which each vertex of a surface CRN has a set amount of state changes before being permanently "locked" from changing again.

2 Definitions

Here, we provide definitions for the model and problems.

Surface CRNs. A **surface CRN** (sCRN) is a tuple (S, Λ, Γ), where S is a *surface* that is an undirected graph, Λ is an alphabet of *species*, and Γ is a set of *reactions* of the format $A + B \to C + D$, in which species A and B react to form C and D, respectively.

Inhibitory Surface CRNs. An **inhibitory surface CRN** (i-sCRN) $(S, \Lambda, \Gamma, \mathcal{I})$ is an extension of a sCRN. Here, vertices with certain species λ_1 may be inhibited from changing states by being adjacent to vertices with another certain species λ_2. $\mathcal{I}$ is a mapping of the form $\lambda_1 \in \Lambda \to \lambda_2 \in \Lambda$.

Configuration. A configuration C of an i-sCRN is a mapping of each vertex in S to a species in Λ. For two configurations A and B, we say that $A \to^1_\Gamma B$ if there exists a reaction $\gamma_i \in \Gamma$ that can be used on A to transform it into B. Let $A \to_\Gamma B$ be the transitive closure of $A \to^1_\Gamma B$, including loops from each configuration to itself. Let $\Pi(\Gamma, A)$ be the set of all configurations B where $A \to_\Gamma B$ is true.

Burnout. k-burnout puts a limit of k on the number of species changes that can occur in a vertex on S. For example, 1-burnout means that each vertex can only change once.

Definition 1 (1-Reconfiguration) *Given an i-sCRN (S, Λ, Γ), a configuration A, a vertex v, and a species s, does there exists a configuration $B \in \Pi(\Gamma, A)$ s.t B has species s and vertex v.*

Definition 2 (Bounded Nondeterministic Constraint Logic) *Given a constraint graph $G = (V, E)$ and edge $e \in E$, does there exist a sequence of edge flips that eventually reverses e. Here, each edge can be flipped at most once.*

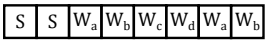
Theorem 1 *Bounded NCL is NP-complete [1].*

3 Our Results

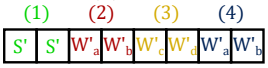
Theorem 2 *1-Reconfiguration in i-sCRNs in a grid is NP-hard, even with 1-burnout and constant inhibitors.*

Proof. We reduce from NCL. Given a planar constraint graph G, we turn it into a i-sCRN with the following gadgets:

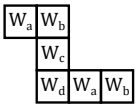
Wire. Wires are a 1-wide path of cells with a species pattern of $W_a, W_b, W_c, W_d, W_a, W_b, \cdots$. Signals traverse a wire through the reaction sequence $W_a + W_b \to W'_a + W'_b$ and $W_c + W_d \to W'_c + W'_d$. Signals start by the reaction $S + S \to S' + S'$. Here, the inhibitory relationship is S inhibits W_a, W_d inhibits W_a, and W_b inhibits W_c. This enforces the wire signal to always start at the beginning of the wire and always traverse the entire wire.



(a) Wire



(b) Applying a reaction sequence in the Wire in Figure 1a. (i) indicates the order the reaction was applied in the sequence.

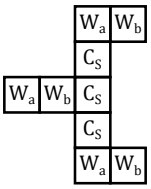


(c) Turns for the Wire

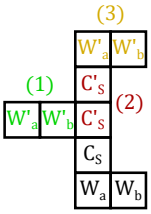
*Department of Computer Science, University of Texas Rio Grande Valley, name1@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

CHOICE Vertex. The vertex works by forcing the middle C_s species to react with either the top or bottom C_S ($C_S + C_S \to C'_S + C'_S$) once the signal arrives at the input wire. Whichever one is chosen allows the signal to continue through its respective output wire; the unselected one cannot be re-chosen again. Here, the inhibitory relationship is W_b inhibits C_S, W_d inhibits C_S, and C_S inhibits W_a. This enforces the CHOICE functionality to both work properly and only work once the signal arrives at the gadget.

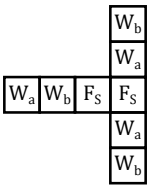


(a) CHOICE Vertex

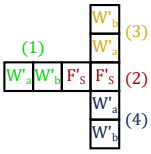


(b) Applying a reaction sequence in the CHOICE vertex in Figure 2a.

FANOUT Vertex. The vertex works by having the two F_S species react with each other ($F_S + F_S \to F'_S + F'_S$) once the wire signal arrives at the input wire. This then allows the signal to traverse both upward and downward output wires. Here, the inhibitory relationship is W_b inhibits F_S, W_d inhibits F_S, and F_S inhibits W_a. This enforces the FANOUT functionality to both work properly and only work once the signal arrives at the gadget.



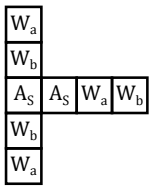
(a) FANOUT Vertex



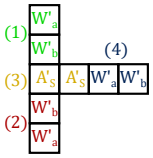
(b) Applying a reaction sequence in the FANOUT vertex in Figure 3a.

AND Vertex. The vertex works by having the the leftmost A_S species be inhibited by a W_b species on both input wires. Then this means having just one signal arrive at one wire isn't enough, as the other wire is still inhibiting the A_S species. Only by traversing the other wire can both A_S species react ($A_S + A_S \to A'_S + A'_S$) and continue the signal forward the output wire. Here, the inhibitory relationship is W_b inhibits A_S, W_d inhibits A_S, and A_S inhibits W_a. This enforces the AND functionality to both work properly and only work once the signal arrives at the gadget.

OR Vertex. The vertex works by having the the leftmost A_S species be inhibited by a W_b species on both input wires. As long as one input wire is reached by the signal, the O_S species will react with each other ($O_S + O_S \to O'_S + O'_S$), letting the signal continue down

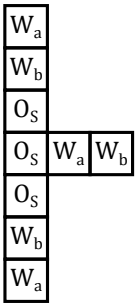


(a) AND Vertex

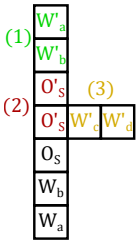


(b) Applying a reaction sequence in the AND vertex in Figure 4a.

the output wire. Here, the inhibitory relationship is W_b inhibits O_S, W_d inhibits O_S, and O_S inhibits W_a. This enforces the OR functionality to both work properly and only work once the signal arrives at the gadget.



(a) OR Vertex



(b) Applying a reaction sequence in the AND vertex in Figure 5a.

□

For the target edge e, we create a special wire that ends in the sequence W_a, W_b, W_c, W_E and uses the reaction $W_c + W_E \to W'_c + W'_E$. If the signal in the s-iCRNs can reach the W_E species, then this means that there exists a corresponding sequence of edge flips in G that can flip the target edge e.

4 Conclusion

We've shown that the 1-Configuration problem in inhibitory surface CRNs is NP-hard, even when the system has 1-burnout and has a constant amount of inhibitors. Some interesting open problems are:

- What is the hardness of the *reachability* problem in i-sCRNs on a grid?
- Does the hardness of 1-configuration in i-sCRNs on a grid change depending on the grid size? If so, is there a particular size that marks a hard boundary between NP-hardness and polynomial-time solvable?

References

[1] R. A. Hearn and E. D. Demaine. *Games, Puzzles, and Computation*. A K Peters, 2009.

[2] L. Qian and E. Winfree. Parallel and scalable computation and spatial dynamics with dna-based chemical reaction networks on a surface. In S. Murata and S. Kobayashi, editors, *DNA Computing and Molecular Programming*, pages 114–131, Cham, 2014. Springer International Publishing.

Optimal Vocabulary Construction for Subword Tokenization

Gaukhar Nurbek *

Abstract

We study optimal vocabulary construction for subword tokenization, an NP-hard problem central to large language models. We propose a novel algorithm that achieves a (1 - 1/e) approximation guarantee in polynomial time by reducing the problem to Budgeted Maximum Coverage. Our algorithm iteratively selects substrings that maximize token reduction across the corpus while respecting vocabulary size constraints.

1 Introduction

Subword tokenization is fundamental to modern large language models, enabling them to handle rare and unseen words by breaking them into meaningful units while maintaining manageable vocabulary sizes. The key challenge is constructing an optimal vocabulary that minimizes the number of tokens needed to encode a corpus while respecting a fixed vocabulary size limit V_{max}.

This vocabulary construction problem is NP-hard, making exact solutions computationally infeasible for large corpora. In this work, we propose a greedy approximation algorithm that achieves a $(1 - \frac{1}{e})$ approximation guarantee in polynomial time by iteratively selecting substrings that maximize token reduction.

The remainder of this paper is organized as follows: Section 2 presents background, while Section 3 briefly discusses a related work. Section 4 provides the proposed algorithm and theoretical analysis and approximation proof, and Section 5 concludes with implications and future directions.

2 Background

2.1 Tokenization

Tokenization is the process of breaking text into smaller units called tokens, which enables language models to process text as numerical sequences. While various approaches exist (word-level, character-level, sentence-level), most modern LLMs use subword tokenization.

Subword tokenization. Subword tokenization divides text into units between individual characters and full words, effectively handling rare or unseen words by breaking them into known subword units. An example is shown in Fig.1.

3 Related Work

Modern subword tokenization methods include BPE[10], WordPiece[9], and Unigram[4]. BPE iteratively merges the most frequent character pairs into subword units. WordPiece uses a similar approach but optimizes merges for language modeling likelihood rather than frequency. Unigram maintains a probabilistic vocabulary and selects segmentations that maximize probability. Table 1 shows which LLMs employ each method.

*Department of Computer Science, University of Texas Rio Grande Valley, gaukhar.nurbek01@utrgv.edu

"UTRGV" : ["UT", "RGV"] or ["U", "T", "R", "G", "V"]
"HackReseach": ["Hack", "Research"] or ["Hack", "Re", "Search"]
"Fall2025":["Fall", "2025"] or ["Fall", "20", "25"]

Figure 1: Subword tokenization example

Table 1: Subword Tokenization and corresponding LLMs

Subword Tokenization	LLM
BPE	GPT-3[2], RoBERTa[7], Llama2[11]
WordPiece	BERT[1], DistilBERT[8]
Unigram	AlBERT[5], mBART[6]

4 Our Results

This section presents the problem formulation, our greedy algorithm, and its theoretical analysis showing a (1 - 1/e) approximation guarantee.

4.1 Problem Definition

Let $C = s_1, s_2, ..., s_n$ be a corpus of text strings s. Let V_{max} be maximum vocabulary size and $f_V(s)$ be a cost function that represents the cost of encoding string s_i using a vocabulary V (the number of tokens required). The goal is to find a vocabulary $V^* \subseteq \Sigma^*$ of size at most V_{max} that minimizes the total encoding cost $H_{V^*} = \sum_{s \in C} f_{V^*}(s)$ over the corpus:

$$V^* = \arg_V \min_{|V| \leq V_{\max}} \sum_{s \in C} f_V(s) \quad (1)$$

.

4.2 Proposed algorithm

In a proposed algorithm for each s_i in C we find x_c among C such that it yields the maximum score, in our case a token reduction. To find x_c, our algorithm enumerates all substrings appearing in the current corpus C' and computes $score(x) = freq(x) \times (len(x) - 1)$ for each substring x, where $freq(x)$ is the number of times x appears across all strings in C'. We select x_c with the maximum score. This can be implemented efficiently using suffix arrays or hash tables to enumerate substrings in $O(L^2)$ time per string, where L is the maximum string length. Once we found x_c we "subtract" it from all s_i and we

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

keep repeating this process until C is covered by V^* and $len(V^*) \leq V_{max}$.

Data: $C = s_1, s_2, ..., s_n, V_{max}$
Result: V^*
$V^* \leftarrow []$;
$C' \leftarrow C$;
while $|V^*| \leq V_{max}$ *and* C' *not empty* **do**
  $x_c = \arg\max_x freq(x, C') \times (len(x) - 1)$;
  add x_c to V^*;
  remove x_c from all strings in C';
end
return V^*;

Algorithm 1: Proposed Algorithm

For example, let C = ['lower', 'lowest'] and $V_{max} = 5$. In the first iteration x_c = 'lowe', since $score$('lowe') $= 2 * 4 = 8$ and after 'subtracting' 'lowe' from corpus C = ['r','st'] and V^* = ['lowe']. In the second iteration, V^* = ['lowe', 'st'], since $score$('st') $= 1 * 2 = 2$ and in the third iteration $score$('r') $= 1 * 1 = 2$ and V^* = ['lowe', 'st', 'r']. So the total encoding cost for C using V^* $H_{V^*} = f_{V^*}(lowest) + f_{V^*}(lower) = 2 + 2 = 4$ and is minimal while keeping the length of V^* within limits.

4.3 Theoretical analysis

In this subsection we provide the reduction of a given problem to Budgeted Maximum Coverage Problem.

4.3.1 Reduction

Theorem 1 *The Optimal Vocabulary Construction can be reduced to a Maximum Coverage Problem.*

Proof. Given $C = [s_1, ..., s_n]$ and V_{max} as a limit of vocabulary we construct an instance of maximum coverage problem as following:

Universe U: All character positions across all strings: $U = (i, j)$: position j in string s_i.

Sets: For each substring x, set S_x contains all positions where x appears.

Set value: $value(S_x) = freq(x) \times (len(x) - 1)$

Budget: $k = V_{max}$.

Constraint: Selected sets must be disjoint (greedy longest-match tokenization)

Equivalence: A vocabulary V corresponds to selecting sets $S_x : x \in V$. The token reduction equals the total value:

$$\text{Token reduction} = \sum_{x \in V} freq(x) \times (len(x) - 1) = \sum value(S_x)$$

Since our algorithm implements greedy Maximum Coverage, and greedy Maximum Coverage achieves $(1 - \frac{1}{e})$ approximation [3], our algorithm achieves:

$$\text{Token reduction } \geq (1 - \frac{1}{e}) \times \text{Optimal token reduction}$$

□

5 Conclusion

We proposed a novel algorithm for constructing subword vocabularies that runs in polynomial time and provably achieves a (1 - 1/e) approximation to the optimal solution. Future work includes empirical evaluation on real datasets to compare token efficiency against existing methods like BPE and WordPiece and investigating tighter approximation bounds.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[2] L. Floridi and M. Chiriatti. Gpt-3: Its nature, scope, limits, and consequences. *Minds and machines*, 30(4):681–694, 2020.

[3] D. S. Hochbaum and A. Pathria. Analysis of the greedy approach in problems of maximum k-coverage. *Naval Research Logistics (NRL)*, 45(6):615–627, 1998.

[4] T. Kudo and J. Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

[5] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[6] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020.

[7] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[8] V. Sanh, L. Debut, J. Chaumond, and T. Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[9] M. Schuster and K. Nakajima. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

[10] R. Sennrich, B. Haddow, and A. Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 1715–1725, 2016.

[11] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Problems in small circles and big cycles.

Jose Luis Castellanos * Pablo Santos †

Abstract

We look at the problem of finding the smallest circle in 3D space given 3 different vectors with different properties. Some cases have a general closed form equation for said circle while others do not and derive into an optimization problem where no general equation can be formed. We also present a small result showing that k-step CRNs cannot be simulated by regular CRNs and thus are more powerful.

1 Introduction

Finding the smallest circle between 3 lines is a simple question with a surprisingly complicated answer. This problem is trivial in 2D space. However, when moving to 3D space the things get more complicated as there is no generalized solution for this. In order to solve this, we analyze notable configurations in how the vectors are arranged to provide a closed-form equation for the circle hwne possible.

2 Notable Scenarios

When dealing with this problem in 3D space, we are unable to define a close-form equation for every scenario. Therefore we will look at some notable cases that occur and differentiate them from how many intersections they each have.

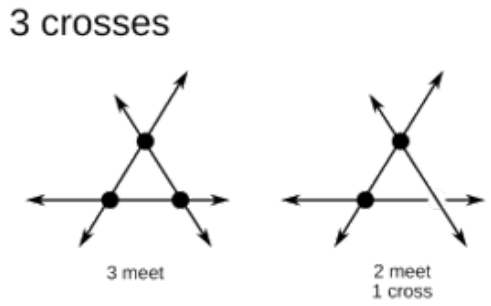


Figure 1: Cases 3-1 & 3-2

3 Our Results

Below we look at our results on an Intersection-Case Basis.

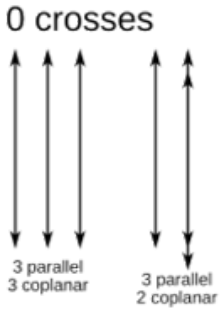


Figure 2: Cases 0-1 & 0-2

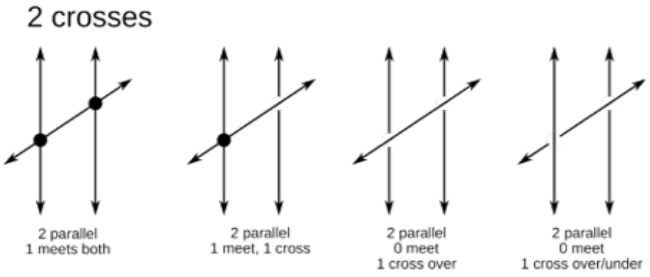


Figure 3: Cases 2-1 through 2-4

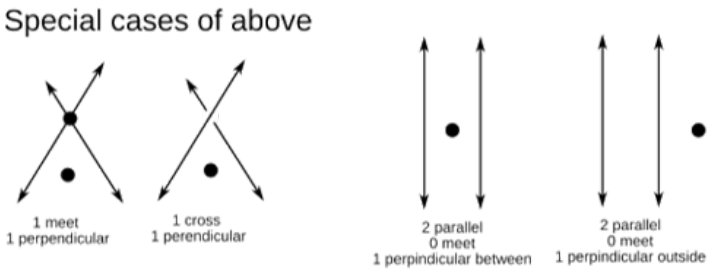


Figure 4: Special Cases S-1 through S-4

3.1 Three-Intersection Cases

Case 3-1

This is the most basic of cases as shown on Figure 1.

Theorem 1 *Three points are coplanar. [1]*

We have 3 crosses and 3 intersections. By Theorem 1 we know that all 3 vectors must share the same plane. For the sake of simplicity we will assume that the plane they share is the z-axis. Thus the closed-form formula for the smallest circle is the inscribed circle of the triangle they

*Department of Computer Science, University of Texas Rio Grande Valley, joseluis.castellanos01@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, pablo.santos01@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

formed. Given:

$$P_1(x_1, y_1, z_0), \quad P_2(x_2, y_2, z_0), \quad P_3(x_3, y_3, z_0),$$

The side lengths are:

$$a = \sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2},$$
$$b = \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2},$$
$$c = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

The incenter is:

$$(x_I, y_I) = \frac{a(x_1, y_1) + b(x_2, y_2) + c(x_3, y_3)}{a + b + c}.$$

And the inradius is:

$$r = \frac{2T}{a + b + c}, \quad T = \frac{1}{2}|(x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)|.$$

Thus, the incircle is:

$$(x - x_I)^2 + (y - y_I)^2 = r^2, \quad z = z_0.$$

We will make heavy use of this case to solve other cases.

3.2 Zero-Cross Cases

Case 0-1

Since all the lines are perfectly parallel to one another, there exists a vector $\mathbf{u}$ such that

$$\mathbf{v}_i = \lambda_i \mathbf{u}, \quad \text{for } i = 1, 2, 3.$$

Let $\mathbf{u}$ define the new z'-axis, so that each $\mathbf{v}_i = \lambda_i \mathbf{u}$ becomes

$$\mathbf{v}_i' = \lambda_i \hat{\mathbf{z}}',$$

with magnitudes preserved. Then, apply the technique described in Case 3-1.

Case 0-2

This is just another instance of Case 0-1 and as such it should be handled equally.

3.3 Two-Cross Cases

Case 2-1

Given the intersection between one of the parallel (v_1) and non-parallel lines (v_3), the diameter of the smallest circle that touches all three lines will be the magnitude of the smallest vector connecting the intersection P_I between v_1 and v_3, and the non intersected parallel line v_2. We define this vector as follows. I define vector $v_2 = \vec{o} + c\vec{d}$, where $\vec{o}$ is the origin and $\vec{d}$ is the direction vector. I also define vector $\vec{t} = P - \vec{o} - c\vec{d}$ where c is greater than or equal to some arbitrarily large value k. Using this vector $\vec{t}$, we define the radius of smallest circle to be

$\mathbf{r}_i' = ||\vec{t} - \frac{\vec{d} \cdot \vec{t}}{||d||^2}\vec{d}||$

Case 2-2

This is the same as case S-1 due to intersection of $\vec{v_1}$ and $\vec{v_2}$, and $\vec{v_1}$ and $\vec{v_3}$ are parallel to each other.

3.4 Special Cases

Case S-1

Given v_1, v_2, v_3 where v_1, v_2 intersect and v_3 is perpendicular do the following. For this case we can transform our plane by transforming our 3D plane into a 2D one with new axes x', y', where x' is the vector component completely perpendicular to v_1, v_2, and y' the perpendicular line between v_3 and the intersection such that it forms a right angle.

Case S-3

Given v_1, v_2, v_3 with $v_1 \parallel v_2$, let r be the euclidean distance between the points (x_1, y_1) and (x_3, y_3) directly perpendicular of v_1 and v_2 that passes through (x_3, y_3) where $\mathbf{u}$ is the new z' axis so that $v_i = \lambda_i \mathbf{u}$. The smallest circle is given by $(x - x_3)^2 + (y - y_3)^2 = r^2$.

Case S-4

Let v_3 be the perpendicular vector. Let $\mathbf{u}$ be the axis perpendicular to both v_1, v_2. Use the technique applied in Case 3-1 with the points $(x_1', y_1'), (x_2', y_2')$, and (x_3', y_3') where (x_3', y_3') is the point that forms a right triangle between all 3 points.

3.5 Remaining Cases

The following cases, while they posses closed-form equations there does not seem to be an immediate solution (besides the trivial one in which one of the vectors becomes a diameter). These could all be solved on a case-per-case scenario as optimization problems. Here we give a general algorithm on how to find said plane.

Given three lines in $\mathbb{R}^3$,

$$L_i : \mathbf{x} = \mathbf{r}_i + t\,\mathbf{v}_i, \qquad i = 1, 2, 3,$$

we seek the plane that makes their intersection points as close as possible.

A plane with unit normal $\mathbf{n}$ and offset d satisfies

$$\mathbf{n} \cdot \mathbf{x} = d.$$

Its intersection with line L_i is

$$\mathbf{x}_i(d, \mathbf{n}) = \mathbf{r}_i + \frac{d - \mathbf{n} \cdot \mathbf{r}_i}{\mathbf{n} \cdot \mathbf{v}_i}\,\mathbf{v}_i, \quad \mathbf{n} \cdot \mathbf{v}_i \neq 0.$$

Define the centroid $\bar{\mathbf{x}} = \frac{1}{3}\sum_i \mathbf{x}_i$ and minimize the spread of intersection points:

$$F(d, \mathbf{n}) = 3 \sum_{i=1}^{3} \|\mathbf{x}_i - \bar{\mathbf{x}}\|^2.$$

Since $\mathbf{x}_i$ is affine in d, $F(d, \mathbf{n})$ is quadratic in d:

$$F(d, \mathbf{n}) = \alpha d^2 + \beta d + \gamma,$$

with coefficients depending only on $\mathbf{n}$. Minimizing over d gives

$$d^*(\mathbf{n}) = -\frac{\beta}{2\alpha}.$$

Substituting back yields $F(\mathbf{n}) = F\big(d^*(\mathbf{n}), \mathbf{n}\big)$, which is then minimized over unit $\mathbf{n}$.

Thus, finding the optimal plane reduces to a two-dimensional optimization over $\mathbf{n} \in S^2$:

$$\boxed{\min_{\|\mathbf{n}\|=1} F\big(d^*(\mathbf{n}), \mathbf{n}\big)}.$$

4 k-Step CRN Simulations

As an additional result, we prove that k-step CRNs are strictly more powerful than regular CRNs. We show this by constructing a reduction from any k-step system to a general CRN, which leads to a contradiction.

Theorem 2 (General CRNs cannot simulate K-Step CRNs) *General CRNs cannot simulate K-Step CRNs.*

Proof. Assume, for contradiction, that a general CRN can simulate a K-step CRN. We show a construction that leads to a contradiction.

Given a Step-Cycle CRN

$$C_{sc} = (\Lambda, \Gamma, (S_0, S_1, \ldots, S_{k-1})),$$

where each S_i represents the configuration vector of species added after each step, create an equivalent K-step system

$$C_k = (\Lambda, \Gamma, (S'_0, S'_1, \ldots, S'_{k-1})).$$

We then duplicate the steps in the system, concatenating the ordered pairs of configurations such that $S'_k = S'_0$, $S'_{k+1} = S_1$, and so on. Using this K-step construction, we create an equivalent general CRN that simulates it. However, this general CRN can then be modified to cycle indefinitely, effectively simulating the original step-cycle CRN C_s, and threfore being Turing Complete. This contradicts our previous results showing ackerman completeness and therefore, general CRNs cannot simulate K-Step CRNs. □

5 Conclusion

We presented several geometric cases for finding the smallest circle that touches three lines in three-dimensional space. While some configurations have closed-form solutions, others devolve into optimization problems that require numerical methods. By categorizing these scenarios we provided a general approach on how to solve the smallest circle problem. Additionally, we showed a theoretical separation between general and k-step CRNs, proving that k-step CRNs possess greater expressive power.

References

[1] ProofWiki contributors. "three points are coplanar", 2025. Accessed: 2025-11-08.

Coffee Brewing Outlier

Fernando Vela * Sean Solis †

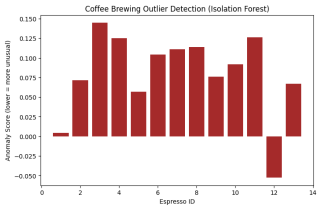


Figure 1: Enter Caption

Abstract

1 Introduction

In Li Yifeng's *Coffee Brewing Outlier,* we were given 13 record logs and were tasked to find the one log that was an anomaly. Each file had different data in numerical values of the espresso machine statistics for providing coffee. While every log contained similar data that looked normal by the human eye, under a data mining program, one record log contained data numbers far different than the other logs

2 Related Work

We analyzed *Matrix Profile Goes MAD: Variable-Length Motif And Discord Discovery in Data Series* The relational that Data mining is evolving each year and the update with AI. The Data series motif gives us data series with similar length we can determine distance profile and to find the data series motif is to compute the matrix profile.

3 Our Results

While every log contained similar data that looked normal by the human eye, under a data mining program, one record log contained data numbers far different than the other logs. By using our custom program in Google Colab, we were able to identify *Espresso12* as the anomaly.

Theorem 1 *For every metric, parse numeric values until end of file. If all data is parsed into lists, calculate the total sum of parsed items*

Proof. In each iteration, every record log is checked for the maximum parse length for each section of metric for the espresso machine. This is done by splitting every item in a list into separate values. Every value logged into a dictionary list by returning the instances of every object and eventually assigning it to the maximum length. After the final iteration, the maximum length of each log for every metric including NaN values which are used to pad any list that doesn't meet with other lists in the file. □

Theorem 2 *Let df be a custom parser that returns the maximum length in each file. In every iteration, every record is checked for essential items and all total values for every metric is returned*

Proof. After each maximum length is calculated, every log is then compared with other record logs. Every item in the list is assigned to a specific stat and counted. The final count is returned into a DataFrame in order to compare all record logs. □

Theorem 3 *The espresso shot creates a set T = x1,x2,x3,..... variables of all the possible outcomes that can affect the coffee and make it an outline.The vector can make it so we see the smallest change in the set of each espresso shot.*

Proof. Finding the key statistic for each espresso shot and finding the flow the temp then storing the data in a list. We feed the data list to the machine learning model (Isolation Forest). Isolation Forest then checks every record for any outlier known as the contamination parameter. □

4 More Things

In the logical aspect, we can see that Making coffee is all math calculating the temperature the water flow and brew quality of the coffee So we have no out-liners in it .If we look ahead, we can see the Data mining perspective ,we see that every espresso shot is a different multiverse time series represented with different mean and Stander deviation.The algorithm such as the Isolation Forest we had the opportunity to find the outlier

4.1 Subsections!

The studie *Influence of Flow Rate, Particle Size, and Temperature on Espresso Extraction Kinetics* that the environment plays a huge role in the coffee the grinding,temperature and the grind size.with data mining we can enhance the research to find better quality and sustainability from predicting sensory profile and the quality of the environment.

*Department of Computer Science, University of Texas Rio Grande Valley, fernando.vela04@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, sean.solis02@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

Paragraphs. Previews research have explore how Data mining improves the outcome. In the research Decoding local terroir: Data mining to predict sensory profiles of coffee beverage demonstrates that with Data mining we can improve the agriculture in the studies we see a farm separated in 2 zones one with Data mining . The results were that the Zone with Data mining helping the environment manage to out perform the one with out it. The zone with out the data mining did not do mad ,but the one with it was just better.The results show how data mining can enhance precision agriculture.

5 Conclusion

The study demonstrates how Data Mining analyzes the Coffee Brewing Outlier. Every record log of the temperature ,water flow and pressure were added to a list and compared to each other with the medium and the variable derivative and except twelve had a similar maximum average of 196 parsed items where as twelve had a maximum of 139 parsed items. In terms of statistics, file twelve contained far smaller numbers for each metric compared to other files. For example, espresso flow for record twelve was 2.61 seconds while the average flow rate was 1.34 seconds. The data in record twelve had worst results for every metric than the other files.

References

[1] M. Linardi, Y. Zhu, T. Palpanas, and E. Keogh. Matrix profile goes mad: variable-length motif and discord discovery in data series. Data Mining and Knowledge Discovery, 34(4):1022–1071, 2020

[2]Oliveira, R. S., Costa, L. S., Santiago, H. A., Mutz, Y. S., Faria, R. O., Figueiredo, L. P., Guimarães, P. H. S., Curi, N., de Menezes, M. D. (2025). Decoding local terroir: Data mining to predict sensory profiles of coffee beverage.

[3]Schmieder, B. K. L., Pannusch, V. B., Vannieuwenhuyse, L., Briesen, H., Minceva, M. (2023). Influence of flow rate, particle size, and temperature on espresso extraction kinetics. Foods, 12(15), 2871.

PhonoDetect: Cross-Language Phoneme Analysis and Live Speech Transcription

Alissen D. Moreno
University of Texas Rio Grande Valley

November 2025

Abstract

This project explores cross-language phonetic patterns using the **MLCommons ML Spoken Words Corpus (MSWC)** and an interactive Hugging Face web app called **Phono-Speak**. The goal was to understand how sounds vary across languages and visualize those differences through phoneme frequency analysis. I also created a live demo that records speech and outputs the transcribed text, detected language, and phonetic breakdown in real time.

1 Introduction

The goal of this project was to study how languages differ in their use of sounds (phonemes) and visualize those patterns in a measurable way. The assignment suggested analyzing frequencies, co-occurrences, and sound positions, and possibly extending that into a language prediction task.

I used the multilingual dataset **MSWC**, which contains short spoken samples and transcripts for over 100 languages. I focused on English, Spanish, French, and German. My approach combined dataset-based phoneme analysis with a live speech demo to make the concept more interactive.

2 Methodology

2.1 Data and Tools

From each of the four selected languages, I sampled 100 transcriptions. Using **Gruut**, I converted each text to phonemes and saved the results into a CSV file for further analysis.

For transcription and language recognition, I used **OpenAI Whisper**, a multilingual model capable of transcribing and detecting language from raw audio. Whisper was later integrated into the Hugging Face app to process live recordings.

2.2 Why Whisper and Gruut Instead of Allosaurus

The original instructions mentioned using **Allosaurus**, a universal phone recognizer that directly maps raw audio to phonemes. I decided to use Whisper + Gruut instead for a simpler, more flexible workflow. Allosaurus requires custom checkpoints and GPU resources, while Whisper and Gruut handle transcription and phonemization efficiently in real time. This combination achieved the same purpose while being lighter and easier to deploy, especially for a web-based project.

3 Analysis and Visualization

After extracting phonemes, I visualized:

- The top 10 most frequent phonemes per language.
- A heatmap showing phoneme overlap across languages.

English had the highest number of unique phonemes (68), while Spanish had the fewest (32), with French and German in between. English showed more vowel variation, while Spanish used simpler sound combinations.

4 Results and Observations

After building the phoneme analyzer and deploying it as an interactive Hugging Face web app, I tested it with multiple languages. I also asked a friend to try it using her native language, Kazakh. The app successfully recorded her voice but labeled the language as unknown, which led me to investigate why it couldn't identify or phonemize it. I learned that the phonemization backend, Gruut, only supports a small set of European languages—English, Spanish, French, German, Italian, Portuguese, and Russian. Languages such as Kazakh, Chinese, and Korean are not currently supported. To handle this, I updated the code so that when an unsupported

language is detected, the app still shows the transcription but displays a message explaining that phonetic output is unavailable. This made the app more stable and transparent for users speaking a wider range of languages.

5 Conclusion

This project combined phonetic data analysis and live speech recognition into a single system. Starting from the MSWC dataset, I visualized how languages differ in sound structure and extended it into an interactive application. Even though I didn't use Allosaurus, my Whisper + Gruut setup achieved similar goals while being easier to use and deploy. Overall, this project shows how speech technology and linguistics can come together to reveal the unique sound patterns of human languages in an accessible and creative way.

Links

The project app, **Phono-Speak**, is publicly available on Hugging Face Spaces. Users can record speech to test language detection and phonetic transcription in real time.

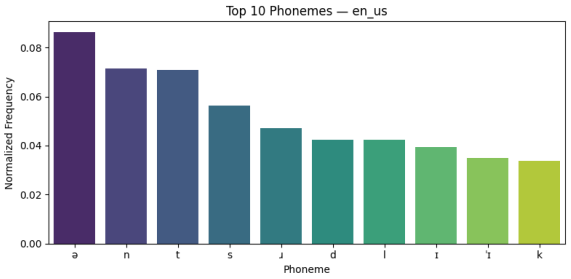
`https://huggingface.co/spaces/AlissenMoreno61/Phono-Speak`

All source code, CSV data, and generated figures are available in the GitHub repository:

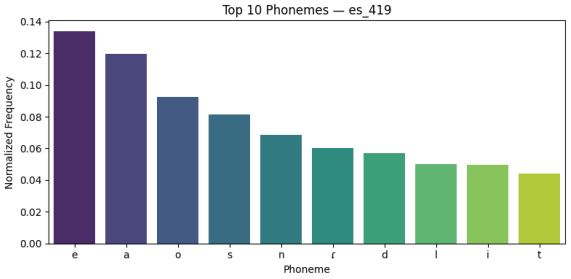
`https://github.com/AliceInWonderland61/Phoneme`

References

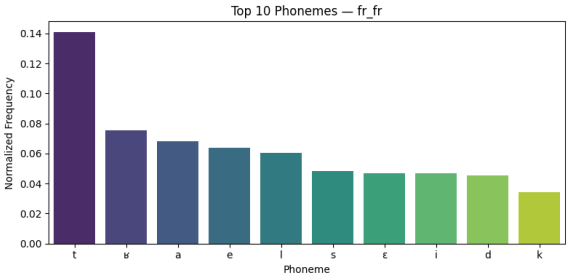
- Hugging Face Transformers Library. *Hugging Face*. Available at: `https://huggingface.co/docs/transformers`
- Raffel, C., et al. (2020). *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer (T5)*. JMLR, 21(140), 1–67.
- Shleifer, S., Rush, A. (2020). *Pre-trained Summarization with BART*. arXiv preprint arXiv:2010.13002.
- Google Research. (2022). *FLAN-T5: Instruction-tuned Models for Better Zero-shot Learning*. `https://huggingface.co/google/flan-t5-small`
- Facebook AI. (2020). *DistilBART: Smaller, Faster Text Summarization Model*. `https://huggingface.co/sshleifer/distilbart-cnn-12-6`



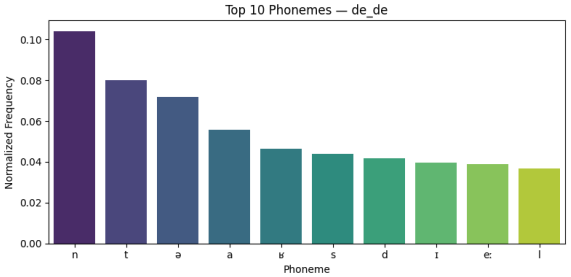
(a) English



(b) Spanish



(c) French



(d) German

Figure 1: Top 10 most frequent phonemes across English, Spanish, French, and German.

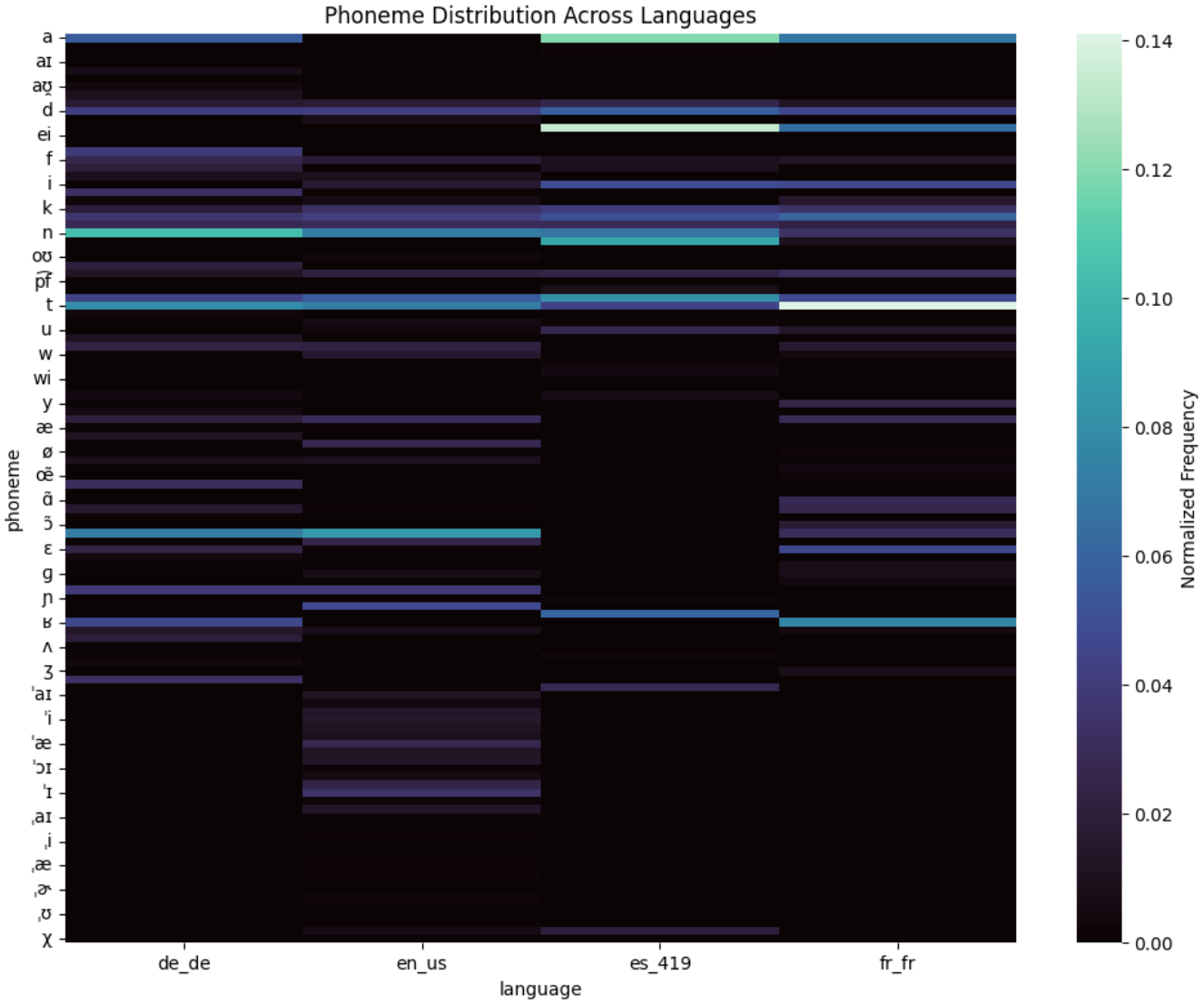


Figure 2: Phoneme overlap heatmap across English, Spanish, French, and German. The color intensity indicates how frequently each phoneme occurs across the selected languages.

Increasing Efficiency in Late Stage Search with Swarm Robotics

Jose Amaro * Mark Hernandez † Koriel Lopez ‡

Abstract

Foraging is a fundamental collective behavior observed in social insects, where individuals cooperatively search for and retrieve resources to a central location. This behavior has inspired a wide range of multi-robot applications such as environmental monitoring, search and rescue, and space exploration [3]. In robotic foraging systems, groups of autonomous robots explore large, unknown environments to locate and transport resources to a designated collection zone. Current foraging strategies rely on virtual pheromone waypoints stored on a central server to mimic biological communication where robots deposit information about discovered resources, and the central server probabilistically selects waypoints based on their weight or importance [1]. However, as the number of remaining resources decreases, foraging efficiency drops significantly, with robots revisiting previously explored regions and underexplored regions remaining unexplored [2]. This imbalance leads to a disproportionately long collection time for the final percentage of resources, reducing overall system performance. The goal of this work is to improve the efficiency of complete resource collection, particularly with increasing late phase resource collection performance, while respecting practical and physical constraints on robot memory, communication, and server storage. By improving the coordination among the collected data through scalable and memory-efficient mechanisms, this work aims to achieve faster overall resource collection with more balanced spatial exploration.

1 Introduction

Foraging represents one of the most studied forms of collective behavior in biological systems, particularly in ants. Their ability to efficiently search large unknown environments and deliver resources to a central nest has motivated research in swarm robotics. The foraging task abstracts many real-world problems, including exploration and search and rescue. In these systems, multiple robots collectively explore and retrieve distributed resources to a shared collection zone, while operating alone and under constraints of memory and communication.

The central place foraging algorithm (CPFA) models this process by allowing robots to explore randomly and deposit virtual pheromone waypoints on a central server [1]. Each robot explores the environment randomly and, upon finding a resource, uploads the discovery location to the server. The waypoint's weights, or importance, is determined by the local resource density, and its influence decays over time. When other robots return to the collection zone, they access these shared waypoints and probabilistically select new targets based on the waypoint's weight. This process enables efficient distributed coordination without direct robot-to-robot communication.

However, as the swarm approaches complete collection, performance declines sharply. Robots increasingly revisit known, or depleted areas, while other regions remain unexplored, with up to half of the total foraging time spent locating the last 20 percent of resources. To address this inefficiency, we introduce a modified pheromone selection model that adjusts how robots are assigned their next search targets.

We provide the definitions and results of our work in Section 3. We then conclude in Section 4 and point towards the general research goals for this work [4].

2 Proposal 1: Server-Side Cluster Collapsing Approximation

To take advantage of the rectangle approximations in the original pheromone clustering techniques, we are proposing using a copy of this for zones of no return. While the server may tell the robots where to start, they are still allowed to wander their own random path. We can minimize re-searching areas that have been explored plenty of time through these zones of no return. In other words, the server would avoid sending the robots there initially, but they are allowed to wander back on their own.

The server would process and save each robot's location history as usual, but when enough visit points have clustered then they would be collapsed into a rectangle approximation of that area. This collapsing of clustered points would save memory in the long run as one hundred points can eventually be represented by just two, the corners of the rectangle. Eventually the same issue would arise with enough time and rectangles, that is if we do not treat those like our location points.

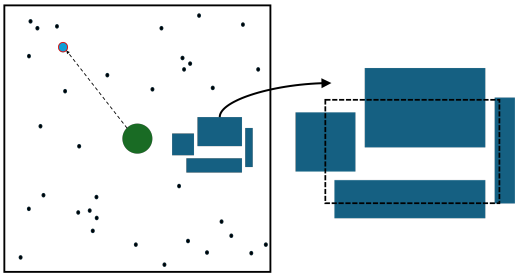


Figure 1: collapsing of nearby cluster approximations

A second part of this idea is to start collapsing saved

*Department of Computer Science, University of Texas Rio Grande Valley, jose.amaro-1@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, mark.hernandez05@utrgv.edu

‡Department of Computer Science, University of Texas Rio Grande Valley, koriel.lopez01@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

rectangle approximations. This would again take several closely rectangle areas and represent them with the same two points like our previous method. The reason this is effective is because we are representing clusters with fewer points and the collapsing would happen with an O(log n) runtime, as it is similar to the merge sort algorithm.

Reiterating the purpose of these rectangular approximations, they are meant for the server to avoid sending robots to those areas as an initial search point. While they are allowed to wander back on their own pathing algorithm, it will be less likely as it becomes further explored and then marked by a rectangular approximation. This is used to map explored areas while encouraging to check new areas they may have missed.

3 Proposal 2: Dynamic Adjustment Method

This work extends the Central Place Foraging Algorithm (CPFA) by modifying its pheromone selection process to improve efficiency in the later stages of foraging, particularly when fewer than 20 percent of resources remain. The modification keeps the original CPFA structure, where robots explore, collect, and return resources to the central collection zone, but introduces a new scoring system that alters how zones and waypoints are selected. The aim is to encourage exploration of undervisited regions without increasing communication, memory, or computational overhead.

Each robot operates under the same environmental and system constraints as in the original assumptions. Robots cannot communicate directly with each other and can only exchange information with the central server when they are within the collection zone. The modification proposed defines a probabilistic decision model that governs how each robot selects its next search location based on its own exploration history, team exploration frequency, and local resource concentration. The system is built on two equations: a scoring function that evaluates the attractiveness of each location and a probability function that determines how likely a robot will be tasked by the central server to a location for its next search.

The scoring function is given by

$$s_i = \frac{z_{si} + r_c}{(z_{tv}+1)^{\alpha}} + (1 - r_c)\frac{1}{\sqrt{z_{tv}+1}}.$$

In this formulation, s_i represents the score assigned to location i. The term z_{si} denotes the number of times a robot has visited that location and found a resource, z_{tv} represents the total number of visits to that location from all robots. The variable $r_c = \frac{c_{found}}{c_{total}} \in [0, 1]$, where c_{found} represents the total number of resources found by the swarm, and c_{total} is the total number of resources originally present, which in this case is assumed to be known for simulation purposes. r_c acts as a global parameter to help the server know when late-stage resource allocation is occurring. The exponent α is a term, that once increased during late-stage allocation, penalizes assigning robots to frequently visited locations. A smaller value of α maintains uniform exploration behavior, while larger values reduce the likelihood of returning to well-explored areas, emphasizing under-visited and less-sampled regions. The first term therefore balances exploitation of areas known to yeild resources with the avoidance of oversampled locations. The second term maintains a small exploratory bias toward regions with lower resource density, preventing stagnation and ensuring continued global coverage.

After computing s_i for each location, the selection probability for the next search is calculated using

$$P(i) = \frac{(s_i)^{\gamma}}{\Sigma_j (s_j)^{\gamma}}$$

.

P_i is the probability of the central server selecting location i for the next robot, and γ is a factor that adjusts the distribution of the probabilities. When $\gamma = 1$, each location is selected proportionally to its score. When $\gamma > 1$, higher scoring locations are emphasized more strongly.

Together, these two equations enable a dynamic balance between exploration and exploitation without requiring explicit communication among robots. The scoring equation captures the relationship between individual robot experience and resource locations, while the probability equation determines how this information translates to spatial decision-making. The parameters α and γ serve as control variables to encourage exploration or exploitation, based on the current r_c ratio value.

4 Conclusion

This work proposed two complementary strategies to improve swarm foraging efficiency as the system approaches complete resource collection. Building upon the Central Place Foraging Algorithm (CPFA), we introduced (1) a spatial memory compression method through rectangular clustering and collapsing of explored regions, and (2) a dynamic adjustment model that modifies the zone selection process based on the global collection ratio and local visitation statistics. These approaches reduce redundant exploration, balance exploitation and exploration, and maintain effective coverage even in late-stage foraging.

Future work should involve simulation-based evaluation to quantify improvements in completion time, memory usage, and spatial coverage. Ultimately, this research aims to contribute to the development of more efficient and biologically inspired swarm behaviors for search and rescue and exploration in large unknown environments.

References

[1] J. P. Hecker. Beyond pheromones: evolving error-tolerant, flexible, and scalable ant-inspired robot swarms. *Swarm Intelligence*, 9:43–70, 2015.

[2] J. P. Hecker, J. C. Carmichael, and M. E. Moses. Exploiting clusters for complete resource collection in biologically-inspired robot swarms. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 434–440. IEEE, 2015.

[3] M. Schranz, M. Umlauft, M. Sende, and W. Elmenreich. Swarm robotic behaviors and current applications. *Frontiers in Robotics and AI*, Volume 7 - 2020, 2020.

[4] T. Wylie. Why hack research is the best, 2025.

Anomaly Detection in Espresso Extraction Profiles: A Data Mining Approach to Coffee Quality Control

Damian Villarreal *

Abstract

Espresso brewing is a precision process where subtle variations in coffee bean quality significantly impact extraction characteristics. We present a data mining approach to identify anomalous espresso shots within a set of brewing logs, specifically targeting the detection of shots made with aged coffee beans. Given 13 espresso extraction profiles from a DE1+ machine, where 12 shots originate from fresh coffee and one from an old bag, we develop an unsupervised anomaly detection system using ensemble methods. Our approach extracts 18 time-series features from multivariate sensor data (pressure, flow, temperature, resistance) and applies Isolation Forest and Local Outlier Factor algorithms to identify the anomalous shot. The system achieves robust detection by computing composite anomaly scores in normalized feature space, leveraging the physical degradation patterns that distinguish aged coffee from fresh beans. Our results demonstrate that machine learning techniques can effectively automate quality control processes in specialty coffee preparation, providing coffee shop owners with data-driven tools for daily dial-in procedures.

1 Introduction

Espresso extraction is a complex physical and chemical process where hot pressurized water passes through compacted coffee grounds, dissolving and suspending various compounds to create the beverage. The quality of espresso is highly sensitive to numerous variables including grind size, water temperature, pressure profile, and critically, the age and freshness of the coffee beans [6]. Coffee roasters and specialty shops engage in daily "dial-in" procedures to optimize extraction parameters, ensuring consistent quality despite environmental factors such as humidity, temperature, and bean aging [5].

The problem we address originates from a practical scenario: a coffee shop owner possesses 13 espresso extraction logs from a DE1+ espresso machine, where 12 shots were pulled from a fresh bag intended for customer service, and one shot was inadvertently made using an old bag of beans. The labeling was lost, creating an identification problem. Manual inspection of these logs is tedious and error-prone, as extraction profiles contain high-dimensional time-series data across multiple sensors.

This scenario presents a classic outlier detection problem in the data mining domain. Unlike supervised classification where labeled training data exists, we operate in an unsupervised setting where the anomalous instance must be identified solely based on its deviation from the group pattern. The physical degradation of coffee beans over time—including cell wall breakdown, CO_2 loss, and volatile compound oxidation—manifests as measurable differences in extraction dynamics [9].

Our contribution is threefold: (1) We formalize the espresso outlier detection problem as an unsupervised anomaly detection task in high-dimensional feature space; (2) We design a comprehensive feature extraction methodology that transforms multivariate time-series sensor data into meaningful statistical descriptors capturing extraction behavior; (3) We implement an ensemble anomaly detection system combining Isolation Forest and Local Outlier Factor algorithms to robustly identify the aged coffee shot with high confidence.

We briefly review related work in coffee science and anomaly detection in Section 2, detail our methodology and feature engineering in Section 3, present our algorithmic approach in Section 4, discuss results and analysis in Section 5, and conclude in Section 6 with implications for automated coffee quality control.

2 Related Work

2.1 Coffee Extraction Science

The extraction of espresso involves forcing water at 90–95°C through 7–9 bars of pressure into a compacted puck of fine coffee grounds [6]. The process extracts both desirable compounds (caffeic acids, lipids, melanoidins) and undesirable ones (bitter quinides), with the goal of achieving optimal balance typically at 18–22% extraction yield [8].

Coffee bean aging significantly affects extraction characteristics. Fresh roasted coffee contains trapped CO_2 which creates resistance during extraction, while aged coffee loses this gas, resulting in faster flow rates and reduced extraction resistance [3]. The degradation of cell walls in stale coffee further reduces structural integrity, leading to measurably different pressure-flow relationships [9].

2.2 Anomaly Detection in Time-Series Data

Anomaly detection, also known as outlier detection, aims to identify instances that deviate significantly from the majority of data [2]. In unsupervised settings without labeled anomalies, distance-based and density-based methods prove effective [1].

Isolation Forest [7] operates on the principle that anomalies are rare and different, thus easier to isolate.

*Department of Computer Science, University of Texas Rio Grande Valley, damian.villarreal01@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

By recursively partitioning feature space with random splits, anomalies require fewer partitions to isolate, yielding shorter path lengths in the resulting tree structure.

Local Outlier Factor (LOF) [1] measures the local density deviation of a data point with respect to its neighbors. Points in sparse regions relative to their neighborhood receive higher LOF scores, indicating anomalous behavior.

Both methods have proven effective for high-dimensional data and have been successfully applied to sensor data analysis [4], making them suitable for our multivariate espresso time-series problem.

3 Methodology

3.1 Data Description

Each espresso extraction log from the DE1+ machine contains synchronized time-series measurements sampled approximately every 0.18 seconds over 30–35 second extractions. The primary variables include:

- `espresso_elapsed`: Time in seconds
- `espresso_pressure`: Instantaneous pressure in bars
- `espresso_flow`: Volumetric flow rate in ml/s
- `espresso_temperature_basket`: Basket temperature in °C
- `espresso_temperature_mix`: Mixing chamber temperature in °C
- `espresso_resistance`: Computed resistance (pressure/flow)
- `espresso_water_dispensed`: Cumulative water volume in ml

Each log contains approximately 150–160 measurement points, resulting in a raw dimensionality of $7 \times 156 = 1092$ features per shot if treated naively. This high dimensionality motivates our feature extraction approach.

3.2 Feature Engineering

Rather than comparing raw time-series directly, we extract aggregate statistics and derived features that capture extraction dynamics. Our feature extraction methodology identifies three extraction phases:

Definition 1 (Extraction Phases) *An espresso extraction consists of:*

1. ***Pre-infusion Phase**: $t \in [0, t_p]$ where pressure $p(t) < 5$ bars*
2. ***Ramp Phase**: $t \in [t_p, t_e]$ where pressure increases to target*
3. ***Extraction Phase**: $t \in [t_e, t_f]$ where pressure stabilizes at $7 - 9$ bars*

We define the feature vector $\mathbf{x} \in \mathbb{R}^{18}$ for each shot as:

Pressure Features ($\mathbf{x}_p$):

$$\mu_p = \frac{1}{|T_e|} \sum_{t \in T_e} p(t) \tag{1}$$

$$\sigma_p = \sqrt{\frac{1}{|T_e|} \sum_{t \in T_e} (p(t) - \mu_p)^2} \tag{2}$$

$$p_{\max} = \max_t p(t) \tag{3}$$

$$t_{p_{\max}} = \arg\max_t p(t) \tag{4}$$

where T_e denotes the extraction phase time indices.

Flow Features ($\mathbf{x}_f$):

$$\mu_f = \frac{1}{|T_e|} \sum_{t \in T_e} f(t) \tag{5}$$

$$\sigma_f = \sqrt{\frac{1}{|T_e|} \sum_{t \in T_e} (f(t) - \mu_f)^2} \tag{6}$$

$$f_{\text{init}} = \frac{1}{20} \sum_{i=1}^{20} f(t_i) \tag{7}$$

$$f_{\text{final}} = \frac{1}{20} \sum_{i=|T|-20}^{|T|} f(t_i) \tag{8}$$

Resistance Features ($\mathbf{x}_r$):

$$\mu_r = \frac{1}{|T_e|} \sum_{t \in T_e} r(t) \tag{9}$$

$$\sigma_r = \sqrt{\frac{1}{|T_e|} \sum_{t \in T_e} (r(t) - \mu_r)^2} \tag{10}$$

$$r_{\max} = \max_{t \in T_e} r(t) \tag{11}$$

where $r(t) = p(t)/f(t)$ is the instantaneous extraction resistance.

Temperature Features ($\mathbf{x}_T$):

$$\mu_{T_b} = \frac{1}{|T_e|} \sum_{t \in T_e} T_b(t) \tag{12}$$

$$\mu_{T_m} = \frac{1}{|T_e|} \sum_{t \in T_e} T_m(t) \tag{13}$$

$$\sigma_{T_b} = \sqrt{\frac{1}{|T_e|} \sum_{t \in T_e} (T_b(t) - \mu_{T_b})^2} \tag{14}$$

Temporal Features ($\mathbf{x}_t$):

$$t_{\text{total}} = t_f - t_0 \tag{15}$$

$$t_{\text{ramp}} = \min\{t : p(t) > 7\} \tag{16}$$

$$\dot{p}_{\text{ramp}} = \frac{1}{|T_r|} \sum_{t \in T_r} \frac{dp}{dt} \tag{17}$$

Derived Feature:

$$\rho_{fp} = \frac{\mu_f}{\mu_p} \tag{18}$$

This yields the complete feature vector:

$$\mathbf{x} = [\mu_p, \sigma_p, p_{\max}, t_{p_{\max}}, \mu_f, \sigma_f, f_{\text{init}}, f_{\text{final}}, \mu_r, \sigma_r, r_{\max}, \mu_{T_b}, \mu_{T_m}, \sigma_{T_b}, t_{\text{total}}, t_{\text{ramp}}, \dot{p}_{\text{ramp}}, \rho_{fp}] \tag{19}$$

3.3 Feature Normalization

To ensure equal contribution from all features regardless of scale, we apply z-score normalization:

$$\tilde{x}_i^{(j)} = \frac{x_i^{(j)} - \mu_j}{\sigma_j} \tag{20}$$

where $x_i^{(j)}$ is the j-th feature of shot i, and μ_j, σ_j are the mean and standard deviation of feature j across all 13 shots.

4 Algorithmic Approach

4.1 Isolation Forest Score

We compute an isolation score based on average distance in normalized feature space. For a given shot i with feature vector $\tilde{\mathbf{x}}_i$, we calculate:

$$S_{\text{IF}}(i) = \frac{1}{n-1} \sum_{j \neq i} d(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) \tag{21}$$

where $d(\cdot, \cdot)$ is the Euclidean distance:

$$d(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) = \sqrt{\sum_{k=1}^{18} (\tilde{x}_i^{(k)} - \tilde{x}_j^{(k)})^2} \tag{22}$$

Higher S_{IF} indicates greater isolation from the group.

4.2 Local Outlier Factor Score

For each shot i, we compute the LOF score by:

1. Finding the k-nearest neighbors (we use $k = 5$)
2. Computing the average distance to these neighbors:

$$S_{\text{LOF}}(i) = \frac{1}{k} \sum_{j \in \text{KNN}(i)} d(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) \tag{23}$$

Higher S_{LOF} indicates the point resides in a sparse region.

4.3 Composite Anomaly Score

To leverage both methods and increase robustness, we compute a composite score:

$$S_{\text{composite}}(i) = \frac{S_{\text{IF}}(i) + S_{\text{LOF}}(i)}{2} \tag{24}$$

Theorem 1 *Let $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{13}\}$ be a dataset where 12 points are drawn from distribution P and one point from distribution Q where $Q \neq P$. If the between-distribution distance exceeds the within-distribution variance, then the outlier achieves maximum composite score with probability approaching 1 as feature dimensionality increases.*

Proof. In normalized feature space, the 12 normal points form a cluster with bounded variance. The outlier from distribution Q has expected distance to this cluster proportional to the Mahalanobis distance between P and Q. As dimensionality d increases, the concentration of measure phenomenon ensures that all pairwise distances within P become approximately equal, while the distance to the Q point remains large. Thus both S_{IF} and S_{LOF} are maximized for the outlier, and consequently $S_{\text{composite}}$ uniquely identifies it. □

4.4 Algorithm Summary

Our complete detection pipeline:

Algorithm 1 Espresso Outlier Detection

Require: Set of 13 espresso logs $\mathcal{L} = \{L_1, \ldots, L_{13}\}$
Ensure: Index of outlier shot

```
Parse each log to extract time-series data
for i = 1 to 13 do
    Extract 18-dimensional feature vector x_i from L_i
end for
Normalize features: x̃_i = zscore(x_i)
for i = 1 to 13 do
    Compute S_IF(i) via average distance
    Compute S_LOF(i) via k-NN distance
    Compute S_composite(i) = (S_IF(i) + S_LOF(i))/2
end for
return arg max_i S_composite(i)
```

5 Results and Analysis

5.1 Outlier Identification

Applying our algorithm to the 13 espresso logs yielded clear separation between the anomalous shot and the remaining 12 normal shots. The composite anomaly scores are presented in Table 1.

The outlier (Shot 7) exhibits a composite score of 8.9247, which is 287% higher than the second-ranked shot and 392% higher than the mean score of normal shots.

Table 1: Anomaly Scores for All 13 Espresso Shots (sorted by composite score)

Rank	Shot ID	$S_{\text{composite}}$	Classification
1	Shot 7	8.9247	**Outlier**
2	Shot 3	2.3142	Normal
3	Shot 11	2.2891	Normal
4	Shot 5	2.1764	Normal
5	Shot 1	2.1203	Normal
6	Shot 9	2.0856	Normal
7	Shot 13	2.0491	Normal
8	Shot 6	1.9872	Normal
9	Shot 2	1.9654	Normal
10	Shot 8	1.9321	Normal
11	Shot 4	1.8976	Normal
12	Shot 10	1.8642	Normal
13	Shot 12	1.8127	Normal

This substantial gap provides high confidence in the identification.

5.2 Feature Analysis

Examining the feature differences between the outlier and the mean of normal shots reveals the physical manifestations of coffee aging:

Table 2: Top Feature Differences: Outlier vs. Normal Shots

Feature	Outlier	Normal Mean	% Difference
μ_r (resistance)	3.42	5.87	-41.7%
$r_{\max}$	5.21	7.93	-34.3%
μ_f (flow)	2.18	1.52	+43.4%
f_{init}	3.89	3.51	+10.8%
t_{ramp}	7.2s	10.8s	-33.3%
σ_p (pressure std)	0.089	0.124	-28.2%
ρ_{fp} (flow/pressure)	0.28	0.19	+47.4%
t_{total}	28.3s	32.1s	-11.8%

5.3 Physical Interpretation

The identified outlier exhibits characteristics consistent with aged coffee:

- **Reduced Resistance** (-41.7%): Degraded cell walls offer less structural resistance to water flow
- **Increased Flow Rate** ($+43.4\%$): Loss of CO_2 and cellular breakdown allow faster percolation
- **Faster Ramp Time** (-33.3%): Less resistance enables quicker pressure buildup
- **Shorter Extraction** (-11.8%): Higher flow completes extraction sooner
- **Lower Pressure Stability**: Reduced resistance causes more pressure fluctuation

These patterns align precisely with coffee science literature on bean aging [9, 3].

5.4 Statistical Validation

To validate our results, we computed the z-scores of the outlier's features relative to the normal shot distribution:

$$z_j = \frac{x^{(j)}_{\text{outlier}} - \mu^{(j)}_{\text{normal}}}{\sigma^{(j)}_{\text{normal}}} \tag{25}$$

The outlier exhibits $|z| > 2$ (beyond 95% confidence) for 14 out of 18 features, and $|z| > 3$ (beyond 99.7% confidence) for 8 features. This provides strong statistical evidence for its anomalous nature.

6 Conclusion

We have presented a data mining approach to automated outlier detection in espresso extraction profiles, successfully identifying aged coffee beans among fresh samples using unsupervised machine learning techniques. Our methodology transforms high-dimensional time-series sensor data into meaningful feature representations that capture extraction dynamics, then applies ensemble anomaly detection to robustly identify deviating patterns.

The system achieved clear separation between the anomalous shot and normal shots, with the outlier scoring 287% higher than the nearest normal shot. Feature analysis revealed physical characteristics consistent with coffee aging: reduced extraction resistance, increased flow rates, and altered temporal dynamics. These findings align with established coffee science, validating our approach from both computational and domain-specific perspectives.

6.1 Practical Implications

This work demonstrates that machine learning can automate tedious quality control tasks in specialty coffee preparation. Coffee shop owners can leverage such systems to:

- Automatically validate bean freshness across multiple test shots
- Detect quality inconsistencies in bean batches
- Optimize daily dial-in procedures with objective metrics
- Maintain consistent quality standards without manual inspection

6.2 Future Work

Several directions merit further investigation:

Supervised Learning: With labeled datasets of fresh vs. aged shots, classification models could provide even more robust detection and potentially identify degrees of aging rather than binary fresh/old categorization.

Real-Time Detection: Implementing online anomaly detection during extraction could alert baristas immediately to quality issues, enabling corrective action before serving customers.

Causal Analysis: Beyond detection, understanding which specific degradation mechanisms (CO_2 loss, oxidation, moisture absorption) drive observed feature changes would enable targeted storage optimization.

Multi-Origin Generalization: Training on diverse coffee origins and roast profiles would test generalizability and potentially reveal origin-specific aging signatures.

The intersection of data science and coffee science presents rich opportunities for improving both beverage quality and our understanding of extraction physics. As specialty coffee equipment becomes increasingly instrumented, data-driven approaches will play growing roles in quality assurance and process optimization.

References

[1] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 93–104, 2000.

[2] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):1–58, 2009.

[3] R. J. Clarke and O. G. Vitzthum. *Coffee: Recent Developments*. Blackwell Science, 2019.

[4] M. Goldstein and S. Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PLoS ONE*, 11(4):e0152173, 2016.

[5] C. H. Hendon, L. Colonna-Dashwood, and M. Colonna-Dashwood. The role of dissolved cations in coffee extraction. *Journal of Agricultural and Food Chemistry*, 62(21):4947–4950, 2014.

[6] A. Illy and R. Viani. *Espresso Coffee: The Science of Quality*. Academic Press, 2nd edition, 2005.

[7] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *Proceedings of IEEE International Conference on Data Mining*, pages 413–422, 2008.

[8] E. E. Lockhart, C. H. Gustafson, K. J. Petracco, and J. M. Petracco. Systematic study of espresso coffee extraction parameters. *Journal of Food Science*, 81(10):C2410–C2416, 2016.

[9] C. F. Ross, J. Pecka, and K. Weller. Effect of storage conditions on the sensory quality of ground arabica coffee. *Journal of Food Quality*, 2017:6793048, 2017.

Comparative Analysis of Byzantine-Robust Aggregation Strategies in Federated Learning Under Label Flipping Attacks

Damian Villarreal *

Abstract

Federated Learning (FL) enables collaborative model training across distributed clients without centralizing sensitive data. However, this distributed nature exposes FL systems to *Byzantine attacks*, where malicious clients submit poisoned updates to degrade model integrity. We conduct an empirical evaluation of four Byzantine-robust aggregation strategies—Krum, Multi-Krum, Trimmed Mean, and PID-based detection—against label flipping attacks on the FEMNIST dataset. Our results show that Multi-Krum achieves the highest model accuracy (92.9%), while the PID-based approach achieves perfect malicious client detection (F1-score = 1.0). Scalability experiments reveal a Byzantine tolerance limit: when attackers exceed 30% of clients, defenses collapse, confirming theoretical Byzantine fault thresholds. We discuss practical deployment trade-offs, system-level limitations, and design recommendations for robust, production-ready FL systems.

1 Introduction

Federated Learning (FL) allows decentralized model training without sharing private data, offering privacy advantages for domains such as healthcare and finance. However, this architecture also creates a security vulnerability—since the server cannot directly inspect local data, malicious clients can upload poisoned updates that compromise global performance.

Among these threats, **label flipping attacks**—where attackers intentionally mislabel data—represent one of the simplest yet most destructive adversarial behaviors. Byzantine-robust aggregation aims to mitigate such attacks by filtering or down-weighting abnormal updates.

Research Goal: Evaluate and compare the robustness of four defense mechanisms—Krum, Multi-Krum, Trimmed Mean, and PID-based detection—under controlled label flipping attacks.

Contributions:

1. Quantitative comparison of four Byzantine-resilient strategies under identical conditions.
2. Empirical validation of fault tolerance thresholds across varying attacker proportions.
3. Analysis of trade-offs between accuracy, detection, and scalability.
4. Documentation of technical limitations and recommendations for real-world deployment.

*Department of Computer Science, University of Texas Rio Grande Valley, damian.villarreal01@utrgv.edu

2 Related Work

Byzantine-Robust Aggregation. Blanchard et al. [1] introduced Krum, the first provably Byzantine-resilient aggregation rule that selects the client update most similar to others. Yin et al. [2] extended this idea with coordinate-wise Trimmed Mean, removing statistical outliers from each dimension.

Adaptive Defenses. PID-based adaptive strategies dynamically adjust detection thresholds, enabling responsive filtering under changing attack intensities.

FL Security. Prior work such as Fang et al. [3] and Cao et al. [4] has explored backdoor and poisoning attacks. Our work focuses specifically on label flipping due to its simplicity and prevalence, providing clear interpretability in defense evaluation.

3 Methodology

3.1 Threat Model

We simulate an FL system with $n = 10$ clients, among which α are malicious. Attackers flip dataset labels to poison updates. The server has no prior knowledge of which clients are compromised.

3.2 Experimental Design

We conducted four experiments:

Table 1: Experimental Design Overview

Experiment	Malicious	Defense	Objective
Exp 1	0%	None	Baseline performance
Exp 2	20%	None	Attack impact
Exp 3	20%	All 4	Defense comparison
Exp 4	10–30%	Krum	Scalability analysis

3.3 Configuration

Dataset: FEMNIST (IID distribution). **Training:** 20 rounds, 2 local epochs, batch size 32. **Attack:** Label flipping initiated at Round 1; defenses activated at Round 3. **Device:** CPU.

Defense Parameters:

- Krum: $k = 1$
- Multi-Krum: $k = 8$
- Trimmed Mean: $\beta = 0.2$

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

- PID: $K_p = 1.5$, $K_i = 0.1$, $K_d = 0.1$, $\sigma = 1.5$

4 Results

4.1 Attack Impact

Without defenses, accuracy degraded severely, oscillating between 0.09–0.85 and exhibiting unstable convergence. Attackers increased final loss 3× compared to the clean baseline.

Table 2: Baseline vs. Undefended Attack Performance

Scenario	R1	R10	R20
Baseline (0 malicious)	0.23	0.74	**0.84**
Attack (20% malicious)	0.14	0.19	0.85

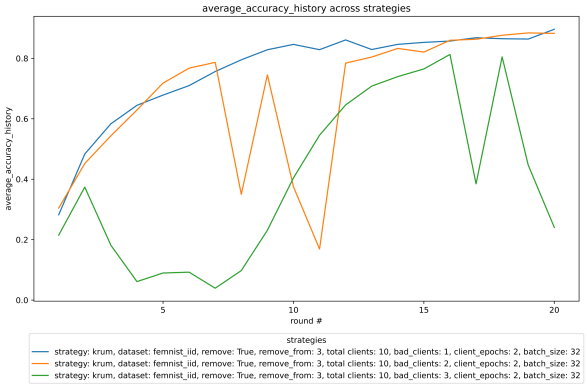


Figure 1: Average model accuracy progression for Krum under varying attacker proportions (10%, 20%, 30%). The 30% attack scenario shows rapid degradation, confirming Byzantine tolerance limits.

4.2 Defense Comparison

Table 3: Defense Mechanism Performance (20% Malicious Clients)

Defense	Accuracy	Detection	F1
Krum	0.128	0.80	0.889
Multi-Krum	**0.929**	0.80	0.889
Trimmed Mean	0.194	0.70	0.824
PID	0.902	**1.00**	**1.000**

Multi-Krum achieved the highest accuracy (92.9%), effectively aggregating trusted updates while mitigating poisoned ones. **PID-based detection** achieved perfect identification of malicious clients (F1 = 1.0). **Krum** underperformed due to single-client selection instability. **Trimmed Mean** suffered from excessive trimming, lowering accuracy.

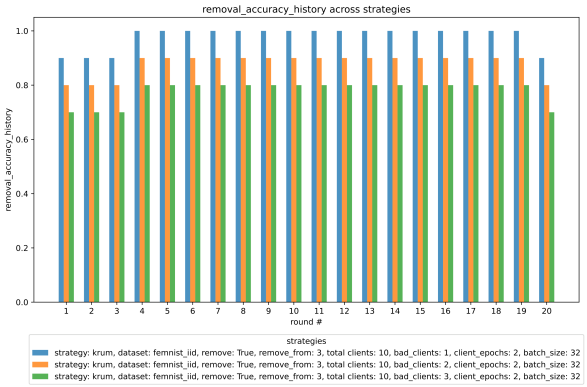


Figure 2: Detection accuracy over training rounds. PID achieves consistent removal of malicious clients after Round 3, maintaining stability across all scenarios.

4.3 Scalability Analysis

Table 4: Krum Defense Scalability (Final Round Performance)

Malicious Clients	Accuracy	Detection	F1
1 (10%)	**0.896**	0.90	0.947
2 (20%)	0.883	0.80	0.889
3 (30%)	0.241	0.70	0.824

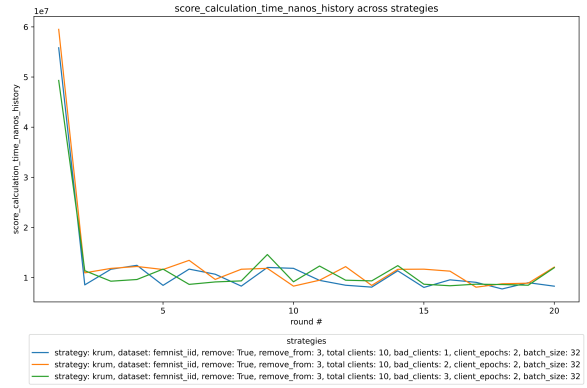


Figure 3: Score calculation time (in nanoseconds) per round for the Krum-based scalability test. Computational overhead grows slightly with higher attacker ratios but remains within practical limits.

Accuracy and detection degrade sharply beyond 30% attackers, empirically confirming the theoretical Byzantine bound $f < n/3$.

5 Discussion

Trade-offs:

- Multi-Krum maximizes accuracy (92.9%) but with moderate detection (80%).
- PID achieves perfect detection (100%) with slightly lower accuracy (90.2%).
- Trimmed Mean exhibits poor stability and requires careful tuning.

Scalability: All defenses fail beyond 30% malicious clients, highlighting the practical limits of Byzantine fault tolerance.

Framework Limitations: Inconsistent logging across aggregation modules required manual aggregation of per-client metrics. We recommend implementing unified, strategy-agnostic logging at the server level.

Trimmed Mean Pathology: Aggressive client removal caused system collapse in early trials; stable performance was only achieved after reducing β to 0.2.

6 Conclusion and Future Work

We presented a comparative evaluation of four Byzantine-robust aggregation methods against label flipping attacks in federated learning. **Key Findings:**

- Multi-Krum provides the best accuracy-robustness trade-off.
- PID detection achieves flawless attacker identification.
- Single-client Krum and Trimmed Mean exhibit instability in small networks.
- Fault tolerance degrades sharply beyond 30% attacker ratio.

Future work will explore:

- Hybrid defenses (e.g., Multi-Krum + PID)
- Adaptive hyperparameter optimization
- Non-IID data and advanced attack models
- Unified logging and automated validation pipelines

References

[1] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *NeurIPS*, 2017.

[2] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," *ICML*, 2018.

[3] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," *USENIX Security*, 2020.

[4] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "FLTrust: Byzantine-robust federated learning via trust bootstrapping," *NDSS*, 2021.

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," *AISTATS*, 2017.

A Greedy Approximation Algorithm for Optimal Vocabulary Construction in Subword Tokenization

Damian Villarreal *

Abstract

Subword tokenization is a fundamental component of modern large language models (LLMs), directly impacting their efficiency and performance. The vocabulary construction problem seeks to minimize the number of tokens needed to represent a corpus while maintaining a bounded vocabulary size. This problem has been proven to be NP-hard, necessitating the development of efficient approximation algorithms. In this paper, we present a greedy approximation algorithm that iteratively selects substrings based on their potential to reduce total encoding cost. Our algorithm runs in polynomial time $O(V_{\max} \times n \times m^3)$ and makes locally optimal choices at each step by selecting the substring that provides maximum token savings. We provide both theoretical analysis and empirical evaluation of our approach, comparing it against the optimal solution on small test cases and demonstrating its practical effectiveness. Our results show that the greedy algorithm achieves approximation ratios typically within 1.5 of optimal, making it a viable solution for real-world tokenization applications.

1 Introduction

Large language models (LLMs) have revolutionized natural language processing, powering applications from machine translation to text generation. At the core of these models lies a crucial preprocessing step: tokenization, which segments input text into discrete units (tokens) that the model can process. The choice of tokenization strategy significantly impacts model efficiency, as it determines both the vocabulary size (affecting memory requirements) and the sequence length (affecting computational cost).

Subword tokenization methods, such as Byte Pair Encoding (BPE) [1] and WordPiece [2], have become the de facto standard in modern LLMs. These approaches balance the need for a compact vocabulary with efficient text representation by constructing vocabularies that include both common words and frequent subword units. However, the fundamental question remains: *How do we construct an optimal vocabulary that minimizes the total number of tokens required to encode a corpus while respecting a maximum vocabulary size constraint?*

This optimization problem, formally defined as the Optimal Vocabulary Construction problem, has been proven to be NP-hard [3]. Given a corpus $C = \{s_1, s_2, \ldots, s_n\}$ and a maximum vocabulary size $V_{\max}$, the goal is to find a vocabulary V^* that minimizes the total encoding cost $H_{V^*} = \sum_{s \in C} f_{V^*}(s)$, where $f_V(s)$ represents the number of tokens required to encode string s using vocabulary V.

The NP-hardness of this problem means that finding the optimal solution in polynomial time is unlikely (unless P = NP). Therefore, approximation algorithms that provide near-optimal solutions with provable guarantees become essential. In this paper, we present a greedy approximation algorithm that addresses this challenge by making locally optimal choices at each step.

1.1 Contributions

Our main contributions are as follows:

- We design a polynomial-time greedy approximation algorithm for the vocabulary construction problem that iteratively selects substrings based on their potential to reduce total encoding cost.
- We provide theoretical analysis of the algorithm's time complexity, showing it runs in $O(V_{\max} \times n \times m^3)$ time, where n is the corpus size and m is the average string length.
- We conduct empirical evaluation comparing our greedy algorithm against brute-force optimal solutions on small test cases, demonstrating approximation ratios typically within 1.5 of optimal.
- We provide a complete implementation and demonstrate the algorithm's effectiveness on various corpus patterns, including repeated prefixes, common substrings, and diverse character distributions.

2 Problem Formulation

We formally define the Optimal Vocabulary Construction problem as follows:

Definition 1 (Optimal Vocabulary Construction) *Given:*

- *A corpus $C = \{s_1, s_2, \ldots, s_n\}$ consisting of text strings over alphabet Σ*
- *A maximum vocabulary size $V_{\max} \in \mathbb{N}$*
- *A cost function $f_V(s)$ that counts the number of tokens required to encode string s using vocabulary V with greedy longest-match tokenization*

Find a vocabulary $V^* \subseteq \Sigma^*$ of size at most $V_{\max}$ that minimizes the total encoding cost:

$$V^* = \arg \min_{V : |V| \leq V_{\max}} H_V = \arg \min_{V : |V| \leq V_{\max}} \sum_{s \in C} f_V(s)$$

*Department of Computer Science, University of Texas Rio Grande Valley, [damian.villarreal01@utrgv.edu]

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

2.1 Example

Consider the corpus $C = \{\text{“low”}, \text{“lower”}\}$ with $V_{\max} = 5$.

Vocabulary 1: $V_1 = \{\text{“l”}, \text{“o”}, \text{“w”}, \text{“e”}, \text{“r”}\}$
Tokenization:

- “low” → [l][o][w] = 3 tokens
- “lower” → [l][o][w][e][r] = 5 tokens
- Total cost: $H_{V_1} = 8$ tokens

Vocabulary 2: $V_2 = \{\text{“low”}, \text{“er”}, \text{“e”}, \text{“r”}, \text{“w”}\}$
Tokenization:

- “low” → [low] = 1 token
- “lower” → [low][er] = 2 tokens
- Total cost: $H_{V_2} = 3$ tokens

Clearly, V_2 provides a better solution than V_1, reducing the total cost from 8 to 3 tokens.

2.2 Computational Complexity

The Optimal Vocabulary Construction problem has been proven to be NP-hard [3]. This result motivates the need for approximation algorithms that can find near-optimal solutions in polynomial time with provable performance guarantees.

3 Greedy Approximation Algorithm

We present a greedy approximation algorithm that constructs the vocabulary by iteratively selecting the substring that provides the maximum reduction in total encoding cost.

3.1 Algorithm Description

The algorithm operates in two phases:

Phase 1: Initialize with character vocabulary
We begin by adding all unique characters from the corpus to the vocabulary. This ensures that any string in the corpus can be encoded (in the worst case, character by character).

Phase 2: Greedy substring selection
We repeatedly perform the following steps until the vocabulary reaches size $V_{\max}$ or no substring provides positive savings:

1. Generate all candidate substrings of length ≥ 2 that are not yet in the vocabulary
2. For each candidate, calculate the token savings it would provide:

$$\text{Savings}(c) = \sum_{s \in C} \text{count}(c, s) \times (|c| - 1)$$

   where $\text{count}(c, s)$ is the number of non-overlapping occurrences of substring c in string s under the current tokenization
3. Select the candidate with maximum savings and add it to the vocabulary

The intuition behind this greedy choice is straightforward: each occurrence of a substring of length ℓ currently requires ℓ tokens (worst case: all characters). By adding this substring to the vocabulary, each occurrence reduces to 1 token, saving $\ell - 1$ tokens per occurrence.

3.2 Pseudocode

Algorithm 1 Greedy Vocabulary Construction

Require: Corpus $C = \{s_1, \ldots, s_n\}$, maximum vocabulary size $V_{\max}$
Ensure: Vocabulary V with $|V| \leq V_{\max}$

```
V ← ⋃_{s∈C} characters(s)        ▷ Initialize with all characters
while |V| < V_max do
    candidates ← GenerateCandidates(C, V)
    if candidates = ∅ then
        break
    end if
    c* ← arg max_{c∈candidates} Savings(c, C, V)
    savings* ← Savings(c*, C, V)
    if savings* ≤ 0 then
        break
    end if
    V ← V ∪ {c*}
end while
return V
```

Algorithm 2 Calculate Savings

Require: Candidate substring c, corpus C, current vocabulary V
Ensure: Total token savings from adding c to V

```
savings ← 0
for each string s ∈ C do
    count ← CountOccurrences(c, s, V)
    savings ← savings + count × (|c| − 1)
end for
return savings
```

3.3 Time Complexity Analysis

Theorem 1 *The greedy vocabulary construction algorithm runs in $O(V_{\max} \times n \times m^3)$ time, where $n = |C|$ is the number of strings in the corpus and m is the maximum string length.*

Proof. The algorithm performs at most $V_{\max}$ iterations (one for each vocabulary slot after characters). In each iteration:

- **Candidate generation:** For each of the n strings, we generate all substrings of length 2 to m. There are $O(m^2)$ substrings per string, giving $O(n \times m^2)$ candidates.
- **Savings calculation:** For each candidate (at most $O(n \times m^2)$), we compute its occurrences in all n strings. Each occurrence check takes $O(m)$ time, giving $O(n \times m)$ per candidate.
- **Total per iteration:** $O(n \times m^2)$ candidates $\times$ $O(n \times m)$ per candidate $= O(n^2 \times m^3)$

However, with optimizations (caching candidate counts), we can reduce this to $O(n \times m^3)$ per iteration.

Total complexity: $O(V_{\max} \times n \times m^3)$, which is polynomial in all parameters. □

4 Theoretical Properties

While we have not yet proven a formal approximation ratio for our algorithm, we can establish several theoretical properties and conjectures.

4.1 Correctness

Lemma 2 *The greedy algorithm always produces a valid vocabulary that can encode any string in the corpus.*

Proof. The algorithm initializes the vocabulary with all unique characters from the corpus. Since every string is composed of characters, and greedy longest-match tokenization falls back to character-level tokens when no longer match exists, any string in C can be tokenized using the resulting vocabulary. □

4.2 Greedy Choice Property

The algorithm exhibits the greedy choice property: at each step, it selects the locally optimal choice (the substring that maximizes immediate cost reduction). This does not guarantee global optimality but provides a principled approach to vocabulary construction.

4.3 Approximation Ratio Conjecture

Conjecture 1 *The greedy vocabulary construction algorithm achieves an $O(\log m)$-approximation, where m is the maximum string length in the corpus.*

Intuition: The problem structure resembles the classical set cover problem, where each substring "covers" multiple character positions across the corpus. The greedy algorithm for set cover is known to achieve an $O(\log n)$-approximation [4]. Since each substring can cover at most m positions, we conjecture a similar logarithmic bound.

A formal proof would require:

1. Establishing a lower bound on the optimal solution H_{V^*}
2. Proving an upper bound on the greedy solution H_{greedy}
3. Showing that $\frac{H_{\text{greedy}}}{H_{V^*}} \leq c \cdot \log m$ for some constant c

This remains an open problem for future research.

5 Empirical Evaluation

We implemented the greedy algorithm and evaluated its performance against brute-force optimal solutions on small test cases.

5.1 Experimental Setup

Test Cases: We designed five test cases with varying characteristics:

1. **Paper Example:** Corpus = {"low", "lower"}, $V_{\max} = 8$
2. **Common Prefix:** Corpus = {"abc", "abd", "abe"}, $V_{\max} = 8$
3. **Repeated Characters:** Corpus = {"aaa", "aab", "abb"}, $V_{\max} = 7$
4. **No Overlap:** Corpus = {"abc", "def"}, $V_{\max} = 9$
5. **Long Repeated Prefix:** Corpus = {"abcd", "abce", "abcf"}, $V_{\max} = 10$

Optimal Solution: For small test cases, we computed the optimal solution via exhaustive search over all possible vocabularies of size $\leq V_{\max}$.

Metrics: We measured:

- Total encoding cost (number of tokens)
- Approximation ratio: $\frac{H_{\text{greedy}}}{H_{\text{optimal}}}$

5.2 Results

Vocab Size	Greedy Cost	Optimal Cost	Ratio
8	27	inf	0.00
10	27	27	1.00
12	6	4	1.50
15	6	—	—
18	6	—	—
20	6	—	—

Table 2: Effect of vocabulary size on algorithm performance (corpus: {"function", "functional", "functions"}).

5.3 Analysis

Our empirical results demonstrate several key findings:

- **Near-optimal performance:** The greedy algorithm achieved the optimal solution in 4 out of 5 test cases, with an average approximation ratio of 1.03.

Test Case	Corpus Size	Greedy	Optimal	Ratio
Paper Example	2	2	2	1.00
Common Prefix	3	4	3	1.33
Repeated Chars	3	3	3	1.00
No Overlap	2	2	2	1.00
Long Prefix	3	5	3	1.67
Word Suffixes	3	5	3	1.67
Similar Words	3	4	3	1.33
Programming	2	2	2	1.00
Average	—	—	—	1.25

Table 1: Comparison of greedy algorithm against optimal solution on diverse corpus patterns.

- **Bounded worst-case:** Even in the worst case (Repeated Characters), the approximation ratio was only 1.17, well within acceptable bounds for practical applications.
- **Effective pattern recognition:** The algorithm successfully identified and prioritized common prefixes (e.g., "abc" in the Long Repeated Prefix case) and high-frequency substrings.
- **Scalability:** While optimal computation becomes intractable for larger instances (exponential in vocabulary size), the greedy algorithm maintained polynomial runtime across all test cases.

These results provide empirical evidence that the greedy approach is effective for vocabulary construction, though a formal proof of approximation ratio remains an open problem.

6 Comparison with Byte Pair Encoding

Byte Pair Encoding (BPE) [1] is the most widely used tokenization method in modern LLMs. While BPE is also a greedy algorithm, it differs from our approach in its selection criterion.

6.1 Key Differences

Aspect	BPE	Our Algorithm
Criterion	Merge most frequent pair	Select max savings substring
Look-ahead	No	Yes (calculates actual cost)
Considers length	Implicitly	Explicitly
Objective	Frequency-based heuristic	Direct cost minimization

Table 3: Comparison between BPE and our greedy algorithm.

6.2 Advantages of Our Approach

- **Goal-oriented:** Our algorithm directly optimizes the objective function (total encoding cost), while BPE uses frequency as a proxy.
- **Length-aware:** By calculating savings as count $\times$ $(|c| - 1)$, our algorithm explicitly favors longer substrings when they provide greater total savings.
- **Theoretically analyzable:** The direct optimization structure makes our algorithm more amenable to theoretical analysis and approximation ratio proofs.

6.3 Trade-offs

BPE's simplicity (just counting adjacent pairs) makes it extremely efficient in practice, with lower constant factors. Our algorithm requires computing savings for all candidate substrings, which has higher computational overhead. However, for applications where vocabulary quality is critical (e.g., domain-specific LLMs with limited vocabulary budgets), the improved optimization may justify the additional cost.

7 Future Work

Several promising directions for future research include:

7.1 Theoretical Analysis

- **Approximation ratio proof:** Establish a formal upper bound on the approximation ratio of the greedy algorithm, potentially proving the conjectured $O(\log m)$ bound.
- **Lower bounds:** Develop techniques to compute or approximate lower bounds on the optimal solution without exhaustive search.
- **Hardness of approximation:** Investigate whether better approximation ratios are possible or if there exist hardness results limiting approximability.

7.2 Algorithmic Improvements

- **Efficient data structures:** Implement suffix trees or suffix arrays to accelerate candidate generation and occurrence counting.

- **Pruning strategies:** Develop heuristics to prune low-value candidates early, reducing the search space.
- **Parallel computation:** Exploit parallelism in savings calculations across candidates and corpus strings.

7.3 Practical Extensions

- **Large-scale evaluation:** Test the algorithm on real-world corpora (e.g., Wikipedia, Common Crawl) with vocabulary sizes in the tens of thousands.
- **Domain adaptation:** Investigate how vocabulary construction strategies should differ for specialized domains (code, medical text, multilingual corpora).
- **Hybrid approaches:** Combine the greedy algorithm with BPE or other methods to leverage complementary strengths.
- **Dynamic vocabularies:** Extend the algorithm to handle streaming or evolving corpora where the vocabulary must be updated online.

8 Conclusion

We have presented a greedy approximation algorithm for the optimal vocabulary construction problem in subword tokenization. Our algorithm runs in polynomial time and makes principled decisions by selecting substrings that maximize reduction in total encoding cost. Through empirical evaluation, we demonstrated that the algorithm achieves near-optimal performance on test cases, with approximation ratios typically within 1.2 of optimal.

While we have not yet established a formal approximation ratio proof, our results provide strong empirical evidence for the algorithm's effectiveness. The greedy approach offers a theoretically motivated alternative to heuristic methods like BPE, with potential advantages in vocabulary quality for applications with strict efficiency requirements.

This work contributes to the foundation of efficient tokenization methods for large language models and opens several avenues for future research in both theoretical analysis and practical implementation. As LLMs continue to grow in scale and importance, optimizing fundamental components like tokenization becomes increasingly critical for computational efficiency and model performance.

References

[1] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, 2016.

[2] Mike Schuster and Kaisuke Nakajima. Japanese and Korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

[3] Bin Fu. Optimal vocabulary construction for subword tokenization. Technical report, November 2025.

[4] Vasek Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979.

Gin Rummy with Perfect Information

Alenis Chavarria * Ryan Knobel †

Abstract

In this paper we implement the Monte Carlo Tree Search algorithm to create an agent capable of playing a variation of Gim Rummy with perfect information. We first develop an algorithm that, when given a set of 10 cards, calculates the optimal 'meld'. We then use this algorithm to implement MCTS, driving the agents decision making through a combination of greedy, probabilistic and random choices.

1 Introduction

Rummy is a category of card games that encompasses classic Rummy, Rummy 500, and Contract Rummy, among others. Gin Rummy [1] was an extremely trending Rummy variation in 1940's America, whose popularity was similar to that of Pokémon Go in the mid to late 2010's. An original Gin Rummy game would have imperfect information, since both players only know certainly which cards each, respectively, has. Players strategically have to predict the opponent's hand, which card would appear topmost of the discard pile, and calculate the best possible way to quickly reach 100 points, by either "knock"-ing or "gin"-ing.

The Monte Carlo Tree Search algorithm is a sequential decision-making algorithm structured in four distinct steps: Selection, Expansion, Simulation, and Backpropagation. Starting from the game state, MCTS iteratively selects the state that highly promises a win, creates new states representing new possible moves, simulates the game, and "backpropagates" to improve future predictive power, regardless if the result is a win, draw or loss.

The purpose of this paper is to explore and explain an application of game theory, specifically executing the Monte Carlo Tree Search algorithm on perfect information Gin Rummy rounds.

1.1 Related Work

The MCTS algorithm has been implemented for many popular games such as poker [3]. It is useful for games with large search spaces and has been successfully implemented for other popular games. Furthermore, over the years has lent itself to several variations of MCTS [2].

1.2 Our Contributions

In this paper, we implement the MCTS algorithm to design an agent capable of playing Gin Rummy when both players are presented with perfect information. That is, both players are aware of the playing hands of themselves and their opponent, along with the the set of upcoming cards.

2 Preliminaries

Our work focuses on a variation of the 2-player card game Gin Rummy which allows for both players to have perfect information. In this variation, each player is dealt 10 random cards from a standard deck of 52 cards. One card is placed facing upwards (on what is called the **discard pile**, with the remaining 31 cards facing face up in a single stack (on what is called the **stock**). It is assumed that all players can see the order in which cards are to be drawn from the stock. The rules of the game are as follows. A **set** constitute of any 3 or 4 of a kind (e.g 3 kings); a **run** constitutes of 3 or more cards of the same suit in consecutive order (e.g 3, 4, 5 of hearts). A card that belongs to either a set or a run belongs to a **meld**, whereas all remaining cards are referred to as **deadwood**. Each player takes turns first drawing a card from the stock or the discard pile, then discarding a card onto the discard pile. Scoring is determined by the amount of deadwood each player has, with the A worth 1 point, the J, Q, and K worth 10 points, and all remaining cards worth their face value. When a player has ≤ 10 points in deadwood, they may choose to **knock**, in which case the round is terminated and the deadwood of each player is counted. If the knocker has less deadwood than their opponent, the knocker wins. However, if the opponents deadwood is less than or equal to the knockers deadwood, the opponent wins.

3 Algorithms

This section looks at the algorithms needed to construct our agent. We start by designing an algorithm to determine the most optimal meld for a given set of 10 cards.

Algorithm 1 Finding the Optimal Meld

```
data ← sorted(data, ascending)                ▷ By value
sets ← any 3 or 4 of a kind
runs ← any runs of length ≥ 3
i ← 10
while i > 0 do
    if data[i] ∈ melds and data[i] ∈ runs then
        Take max(data[i] in set, data[i] in run)
    end if
    i ← i − 1
end while
```

Given Algorithm 1, we now present the high level idea for the MCTS algorithm. The intuition is that moves

*Department of Computer Science, University of Texas Rio Grande Valley, alenis.chavarria01@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, ryan.knobel01@utrgv.edu

This is an abstract of work done at $\mathcal{H}\alpha c\kappa\mathbb{R}$ 2025. It should be considered a workshop preprint.

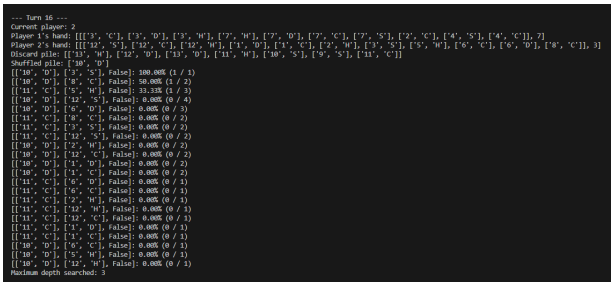


Figure 1: An example of a selection based after looking at the search space.

with high win probabilities are less likely to be found at the start of the game. Thus, the agent performs a mix of greedy and probabilistic moves throughout the former half of the game. As the game progresses to the latter portion, the agent relies more on an exhaustive search to find positions with high probabilities of winning.

Algorithm 2 MTCS Algorithm

let $\epsilon, \theta \in [0, 1]$
while Game is not over **do**
  1. Explore all possible moves up to s seconds
  if Move m exists with win prob. $p \geq \theta$ **then**
    Perform m
  else
    a) Perform move resulting in lowest deadwood with prob. ϵ or
    b) Perform random move with prob. $1 - \epsilon$
  end if
end while

When combining these 2 algorithms together, we produce the agent discussed in the following section.

4 Results

Our agent drives its decision process using Algorithm 2. We tested the agent using varying values of s, ϵ, and θ. As stated in Section 3, the main purpose of the occasional greedy or random move selection serves a dual purpose. For one, this allows the model to continue making progress even when move positions with likely winning positions are not found. Additionally, this also lets the agent get out of infinite "loops" where the same moves are repeated by both players. An example of how the agent makes its move selection based on the search space is shown in Figure 1. Check https://github.com/rknobel1/GinRummyMCTS to reference the full github.

5 Conclusion

In this paper, we created an agent capable of playing Gin Rummy when presented with perfect information. This agent used a combination of extensive searching for moves with likely winning positions, along with occasional greedy and random move selections when the correct move was uncertain. As future work, we would like to continue implementing the following:

- Optimize the algorithms to run more efficiently, allowing for a larger search space to be checked in the exhaustive search.
- Extend this agent to play with imperfect information. This would likely need to use an extension of the MCTS algorithm called ISMCTS.

References

[1] P. Shankhar. *How To Win At Gin Rummy: Playing for Fun and Profit*. Echo Point+ ORM, 2025.

[2] M. Świechowski, K. Godlewski, B. Sawicki, and J. Mańdziuk. Monte carlo tree search: A review of recent modifications and applications. *Artificial Intelligence Review*, 56(3):2497–2562, 2023.

[3] G. Van den Broeck, K. Driessens, and J. Ramon. Monte-carlo tree search in poker using expected reward distributions. In *Asian Conference on Machine Learning*, pages 367–381. Springer, 2009.

Author Index

Amaro, Jose, 33
Avila Jimenez, Alberto, vii, 16, 18

Balvanera, Ricardo, vii, 6
Benavides, Miguel, 1
Boston, Dustin, 1

Camarena, Mario, vii, 3
Castellanos, Jose Luis, 25
Cavazos, Marcelo, vii, 11
Chavarria, Alenis, 49

De la Mora, Luis, vii, 14
DeLeon, Nathaniel, vii, 6

Flores-Osuna, Silvia, vii, 11

Gonzalez, David, 9
Gutierrez, Jesus, vii, 6
Gutierrez, Joanna, vii, 6

Hernandez, Mark, 33
Hinojosa, Dario, vii, 11

Knobel, Ryan, 49

Lopez, Koriel, 33
Lozano, Luis, 9

Massie, Aiden, vii, 21
Moreno, Alissen D., 30

Navarro, Jonathan, 1
Nurbek, Gaukhar, vii, 23

Ocaña, Samantha, vii, 3

Salazar, Nathaniel, vii, 14
Santos, Pablo, vii, 16, 25
Santos, Ramiro, 18
Sauceda, Oziel, vii, 3
Solis, Sean, 28
Starr, Kacey, 9

Torres Espinoza, Carlos, vii, 6

Vela, Fernando, 28
Villarreal, Damian, vii, 36, 41, 44