IMS - Modeling and Simulation
T1: Model in industrial production or agriculture

Documentation

Dmitrii Adamovich (xadamo04)
Veranika Saltanava (xsalta01)

December 2, 2024

Contents

1 Introduction

This document presents a detailed simulation of the production process for processing units in the game **Factorio**, a renowned industrial production simulation game. The goal of this project is to model and evaluate the efficiency of complex production systems using Petri nets and the Simlib simulation library. This work serves as a practical exploration of how simulation techniques can provide insight into real-world-inspired production systems.

The central question driving this work is: **How does modifying a single step in the processing unit production chain affect the efficiency of resource production over time?** This question arises from Factorio's intricate gameplay mechanics, where resource management, interdependencies, and throughput optimization are paramount challenges.

The initial motivation stemmed from Factorio's rich and complex production chain, which closely mirrors the challenges faced in real-world industrial setups. Processing unit production represents a particularly complex system involving interconnected processes such as mining, refining, and chemical manufacturing. Understanding and optimizing this production chain has practical implications, not just in the game but also in broader contexts like manufacturing, logistics, and supply chain optimization.

Our primary objective was to simulate the processing unit production process accurately. To achieve this, we decomposed the system into subsystems, including the production of iron plates, copper cables, and sulfur. Petri nets were employed to capture the dynamic interactions within these systems, and the Simlib simulation library was utilized for implementation. This simulation enables detailed analysis of system bottlenecks, resource allocation strategies, and production efficiency.

The results were validated through rigorous testing and analysis, ensuring that the model reflects expected behaviors and outcomes. Key performance metrics such as production throughput, queue lengths, and resource utilization were measured and evaluated against theoretical predictions. These insights not only contribute to better gameplay strategies in Factorio but also illustrate the utility of simulation in analyzing complex systems.

This documentation is designed for simulation enthusiasts, researchers, and developers interested in industrial optimization. By combining the engaging context of Factorio with robust simulation methodologies, this work bridges the gap between entertainment and practical applications, providing a comprehensive guide to production optimization through simulation.

2 Sources of facts

One of the developers contributed invaluable insights gained from over **1000 hours of gameplay** in Factorio, providing a comprehensive understanding of the game's resource management, production dependencies, and efficiency optimization. This expertise was instrumental in accurately capturing the mechanics and dynamics of Factorio's processing unit production process.

The primary source of factual data for this simulation came directly from Factorio's gameplay. The developer systematically collected and validated data through hands-on experimentation within the game, ensuring that the model accurately reflects the dependencies and timings of the various processes involved. For example:

- The times for mining, smelting, and chemical processing were based on in-game measurements.
- Production rates and resource consumption patterns were verified against gameplay statistics.

While no formal consultation with external experts or authorities in industrial systems was conducted, the project was heavily informed by established simulation practices, including the use of Petri nets for dynamic process modeling and the Simlib library for simulation execution. These tools are recognized in the academic and professional simulation communities for their robustness and applicability to complex systems.

In summary, the key sources of data and expertise for this project are:

1. **Gameplay experience and data collection:** Direct experimentation in Factorio to derive accurate timings, dependencies, and resource flow patterns.

2. **Simulation methodologies:** Established techniques in simulation and modeling, particularly the use of Petri nets and Simlib, to ensure a rigorous and reliable approach.

3. **Factorio Wiki:** The official Factorio Wiki was consulted for detailed descriptions of production processes, item dependencies, and technical details about game mechanics.

This combination of hands-on experience and established simulation practices forms the foundation for the accuracy and reliability of the model developed in this project.

3 Topic

Factorio is a game about designing, building and maintaining factories. After crash landing on an alien planet the player has to launch a rocket back to space in order to win the game. In order to do this, it is necessary to Automate various processes, which is difficult. These can be trees taking up space, rocks, as well as water. logistical problems, hostile locals who are not happy with the growing production. Using the game data we will describe the process of creating a processing unit, as this item is one of the basic and complex items needed to launch a rocket into space, in the current version of the game. The game itself pushes the player to develop. The more complex an item the player wants to create, the more resources and time the player needs to create them. This fact forces the player to improve the production he has already created and makes him think about how he can reuse his existing production items to create new items. At some point, the player has the opportunity to create complex items such as, for example, a processing unit, and is already forced to set up automated production, as the item cannot be created manually. Slowing down during the factory optimisation stages or ignoring problems can lead to malfunctioning of the system and its subsequent destruction by the local hostile inhabitants of the planet - bugs. We have simplified the principle of conveyor operation for visual display of where resources are accumulated in the queue. Simplified logic of the splitter to make its operation clearer and better analyze it. The average value of crude oil production by the pumping machine was taken to understand what to look at and on what to rely on for scaling. The resources in the reserve were also set to zero, because finding new veins with them is not a difficult task, so there is always enough of them. Other elements are as close to the real game as possible.

4 Model Concept

The simulation model is built to represent the intricate processor unit production process from the Factorio video game. Each subsystem in this model is translated into a Petri net diagram to capture the dynamics and dependencies between various resources and processes. These Petri nets serve as the foundation for implementation in the Simlib simulation framework.

4.1 Simulation Model Architecture

The overall architecture of the simulation model is structured as a network of interdependent processes, where each process corresponds to a specific stage in the production of processor units. These stages include mining, smelting, chemical production, and assembly.

The primary components of the model are as follows:

- **Mining operations**: Responsible for extracting iron ore, copper ore, and crude oil.
- **Resource processing**: Includes smelting of ores into plates, refining crude oil into petroleum gas, and chemical processing into sulfur.
- **Assembly operations**: Produce intermediary components like copper cables, plastic bars, and sulfuric acid, ultimately assembling them into advanced circuits and processor units.

4.2 Petri Net Representation

The production process is modeled using Petri nets, where:

- **Places** represent resources or inventory, such as copper plates or petroleum gas.
- **Transitions** depict processes or actions, such as smelting or refining.
- **Tokens** signify the quantity of resources available or being processed.

Each subsystem is detailed below:

1. **Iron Plate Production** (See Figure 1):
   - Describes the transformation of iron ore into iron plates using furnaces.
   - Includes feedback loops for continuous processing based on available iron ore and furnace capacity.
2. **Copper Cable Production** (See Figure 2):
   - Copper plates are processed into copper cables using assembly machines.
   - Includes dual outputs for different downstream needs (e.g., electronic circuits or advanced circuits).
3. **Plastic Bar Production** (See Figure 3):
   - Petroleum gas and coal are processed into plastic bars using chemical plants.
   - Highlights the dependency on both petroleum gas refining and coal mining.
4. **Sulfur Production** (See Figure 4):
   - Combines petroleum gas and water to produce sulfur.
   - Represents the flow of resources through chemical processing units.
5. **Processor Unit Production** (See Figure 5):
   - The final stage combines advanced circuits, plastic bars, and sulfuric acid to assemble processor units.
   - Demonstrates the culmination of multiple resource pipelines into a singular production goal.
6. **General Model Overview** (See Figure 6):
   - A high-level diagram showing the interplay between all subsystems and the flow of resources.

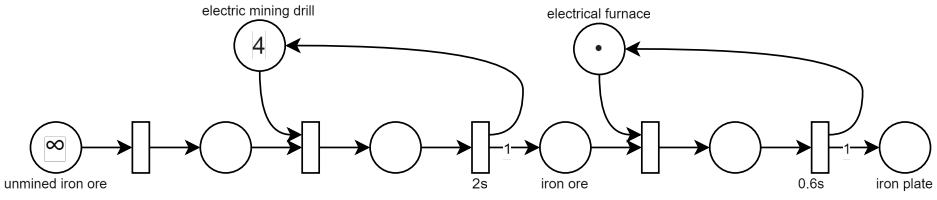


Figure 1: Petri Net for Iron Plate Production.

5 Facts

The game facts used in this work are described below.

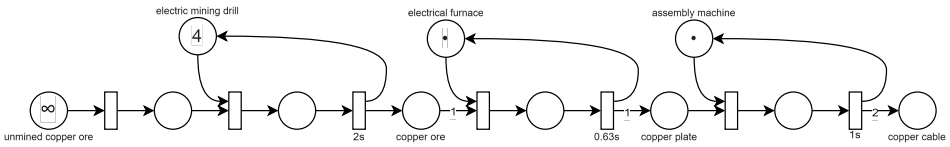


Figure 2: Petri Net for Copper Cable Production.

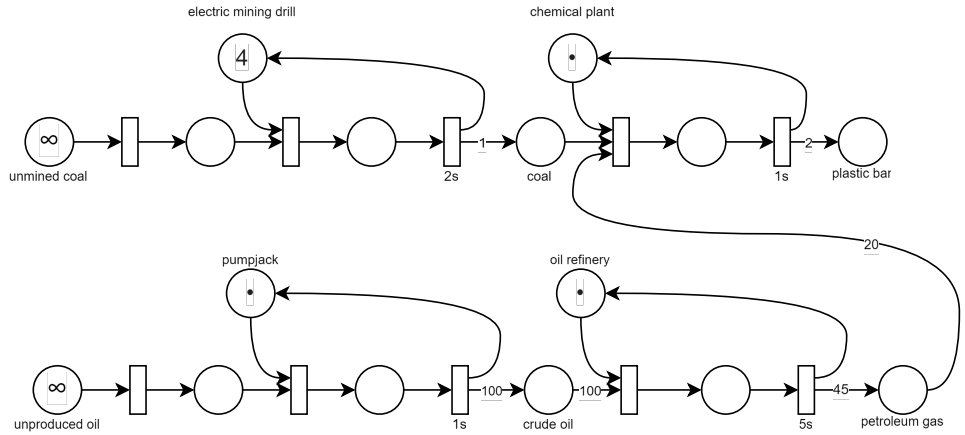


Figure 3: Petri Net for Plastic Bar Production.

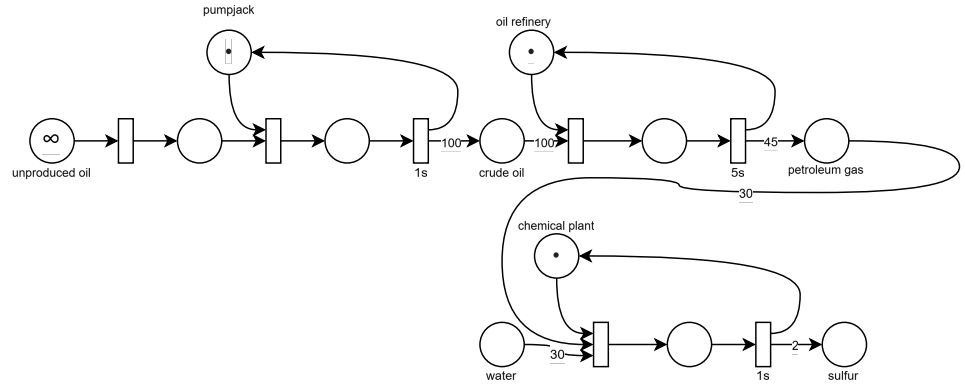


Figure 4: Petri Net for Sulfur Production.

5.1 Assembly Shops

The main production force in the game is the basic assembly shops. Their creation speed has a coefficient of 0.5. The data for the creation time of objects, as indicated on the official page of the game, are considered the basic values. The processor can only be created in a second-level assembly shop, which has a creation rate coefficient of 0.75.

5.2 Electric Furnaces

Our model uses electric furnaces for processing ore into plates. The processing rate of ore was rounded from 0.625 to 0.6, as shown below:

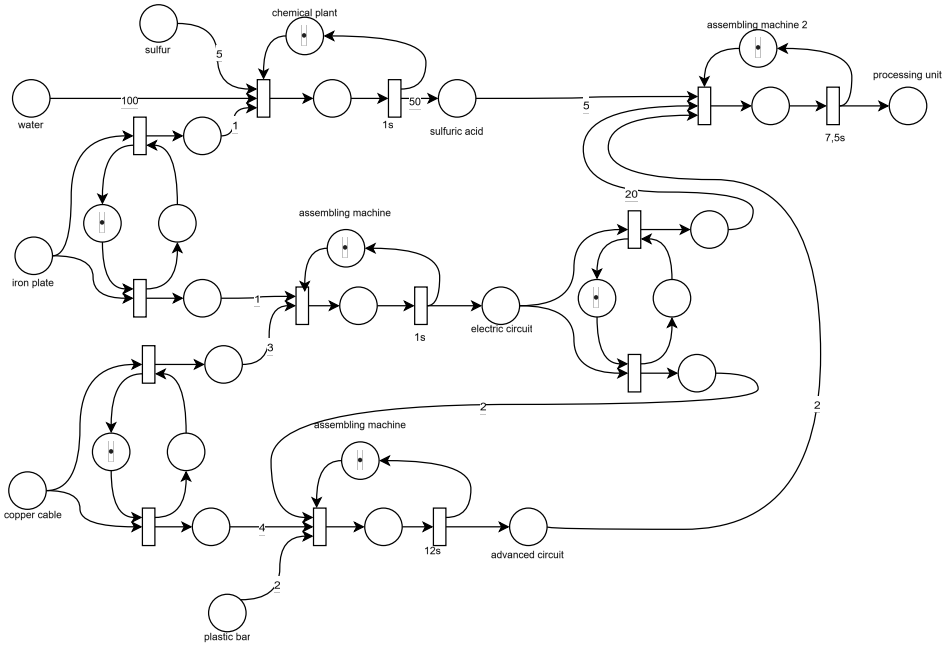


Figure 5: Petri Net for Processor Unit Production.

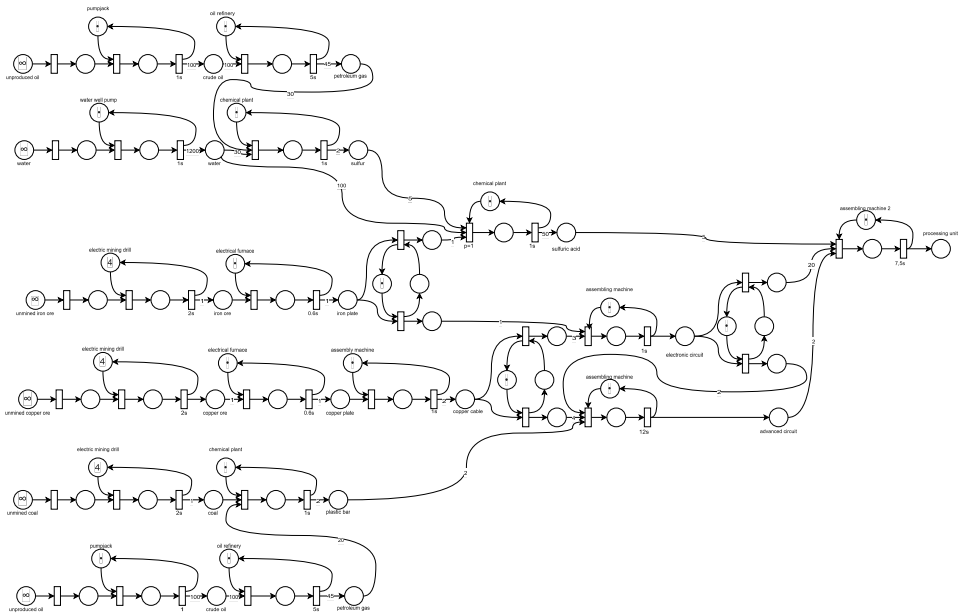
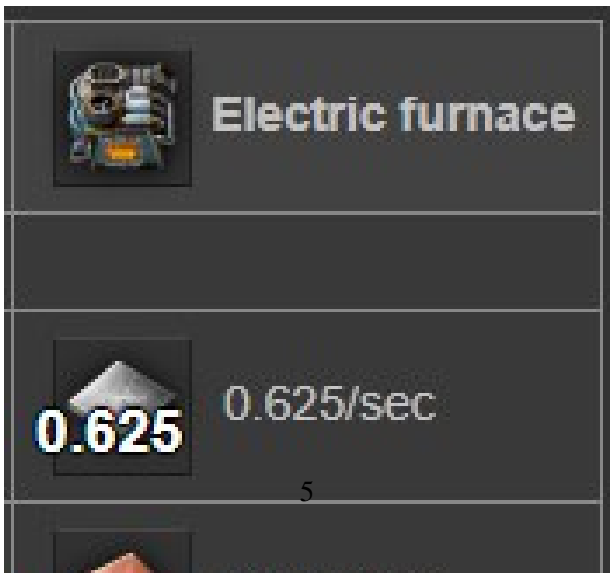
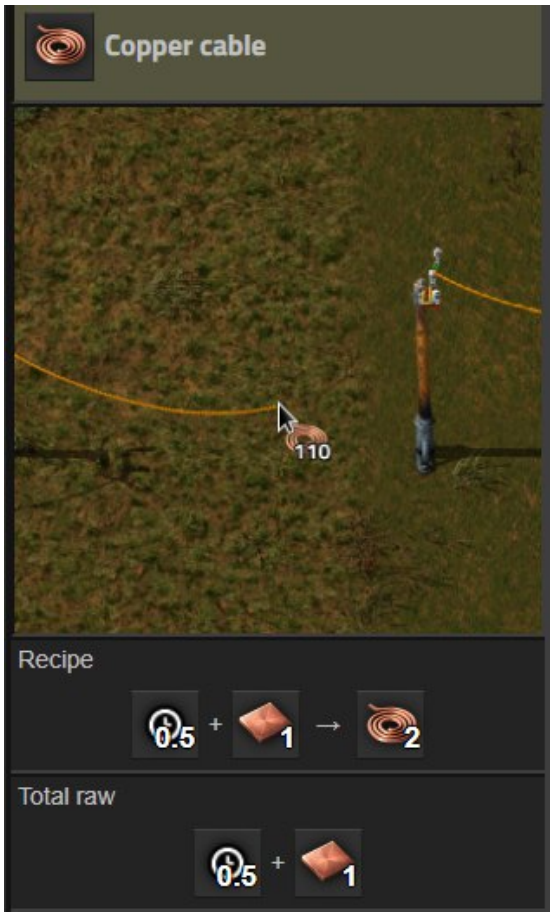


Figure 6: General Model Overview of Processor Unit Production.



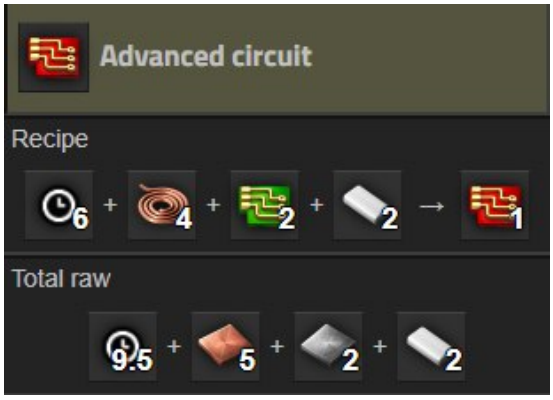
5.3 Recipes for Items Used to Create the Processor

The following recipes were used to create a processor:



Copper Cable

Electronic Circuit



Advanced Circuit

Processing Unit

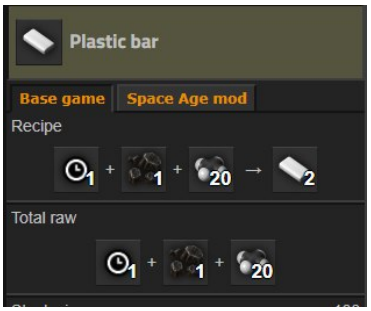
5.4 Oil Refining

The process of refining crude oil into associated gas is carried out in an oil refinery. The operating time and the amount of resources produced correspond to the original game values, as shown below:

Oil Refinery

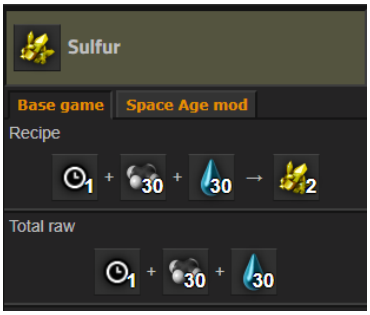
5.5 Chemistry Processes

Processes affecting chemistry, such as the production of plastic bars, sulfuric acid, and sulfur, are fully implemented in the model. The corresponding images are shown below:



Plastic Bar

Sulfuric Acid



Sulfur

5.6 Others

Elements such as an electric drill and a water pump were implemented as standard. The pump jack is implemented with an average amount of crude oil that can be extracted from its source (100 Crud oil/second).

6 Implementation

The C++ implementation of the simulation model uses the SIMLIB library and is mainly based on the concept presented in Section, represented in the form Petri net. This representation can be easily transferred to a program.

6.1 Structure of the program

The project consists of the following source files:

- `main.cpp`: This file contains an implementation of a Petri net modeling the system on the basis of which the experiments are performed.
- `experiment1.cpp`: This file implements a Petri net that displays data about queues and the items stored in them.
- `experiment2.cpp`: The code in this file represents a Petri net designed to show how the production of processing units changes with the increase in missing copper cables.
- `experiment3.cpp`: In this file, a Petri net is implemented with optimization in the form of adding separators to present a more efficient production system and give an idea of possible types of expansion and optimization of processing units production.

6.2 Usage

To run the simulation, a `Makefile` is provided in the root of the project with the following rules:

- `main`: Compiles the `pool.cpp` source file.
- `experiment1`: Compiles the `experiment1.cpp` source file.
- `experiment2`: Compiles the `experiment2.cpp` source file.
- `experiment3`: Compiles the `experiment3.cpp` source file.
- `all`: Compiles all source files and runs them.
- `run`: Runs all binaries after compilation.

- `clean`: Deletes all created binaries.

To use the program, execute the commands according to the rules mentioned above:

```
make <rule>
```

Here is an example of running a specific simulation:

```
make main
./main
```

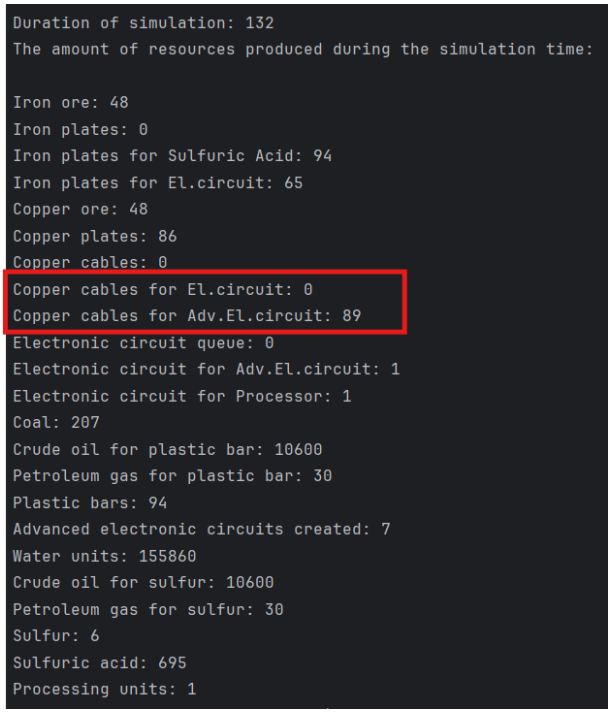
After the simulation is completed, information about queues that can be analyzed will be displayed on the standard output.

7 Simulation Experiments and Results

7.1 Experiment 1: Initial Setup

The first experiment used the initial configuration of the production facilities, including a limited number of drills, furnaces, and assembly machines for each step of the processing unit production process. The simulation was run to observe the baseline performance and identify any bottlenecks in the production system.

7.2 Output Analysis:



The results of this experiment revealed a significant shortage of copper cables, as highlighted in the output logs. This deficiency caused downstream production processes, such as advanced electronic circuits and processing units, to stagnate. The lack of copper cables stemmed from insufficient copper plate production, which limited the ability to create cables efficiently.

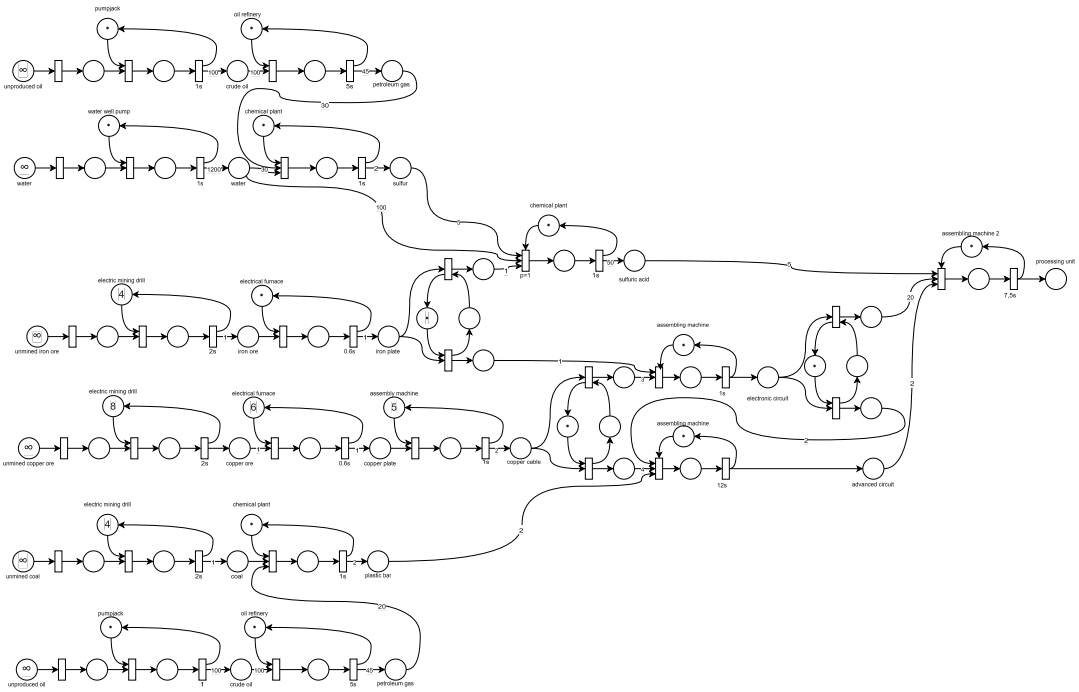
7.3 Experiment 2: Adjusting Copper Cable Production

To address the bottleneck identified in Experiment 1, the number of copper furnaces and assembly machines for cable production was increased. This modification aimed to ensure a consistent supply of copper plates and cables for the production of advanced electronic circuits and processing units.

7.4 Changes Implemented:

- The number of copper drills was doubled to improve the supply of copper ore.
- Additional copper furnaces and assembly machines were introduced to increase the throughput of copper plates and cables.

In the following, we have provided a Petri net that reflects the changes we made:



7.5 Output Analysis:

```
Duration of simulation: 132
The amount of resources produced during the simulation time:

Iron ore: 48
Iron plates: 0
Iron plates for Sulfuric Acid: 94
Iron plates for El.circuit: 0
Copper ore: 8
Copper plates: 2
Copper cables: 0
Copper cables for El.circuit: 193
Copper cables for Adv.El.circuit: 409
Electronic circuit queue: 0
Electronic circuit for Adv.El.circuit: 0
Electronic circuit for Processor: 13
Coal: 207
Crude oil for plastic bar: 10600
Petroleum gas for plastic bar: 30
Plastic bars: 60
Advanced electronic circuits created: 6
Water units: 155860
Crude oil for sulfur: 10600
Petroleum gas for sulfur: 30
Sulfur: 6
Sulfuric acid: 690
Processing units: 2
```

The output of Experiment 2 indicated a marked improvement in the production of copper cables and, consequently, advanced electronic circuits. The increased availability of these components allowed for

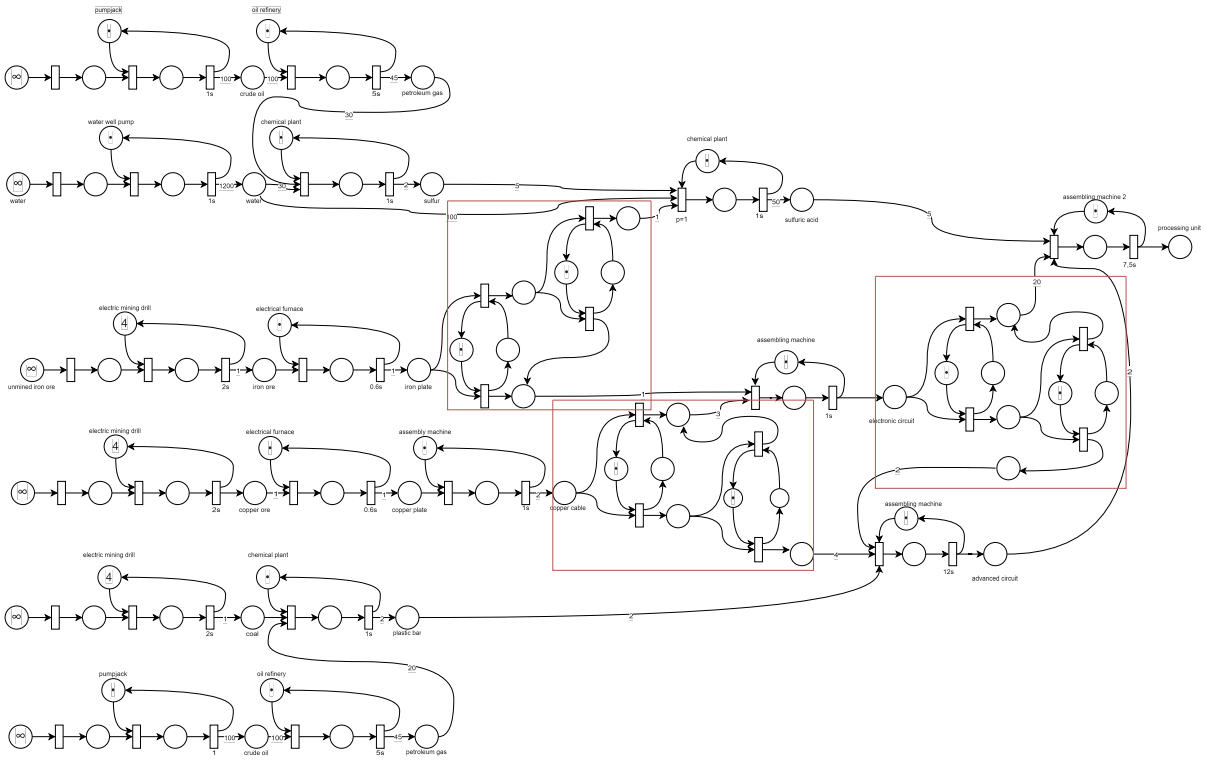
a higher processor production rate, effectively doubling the output compared to Experiment 1. However, the analysis also showed the potential for further optimization in balancing other resources and production facilities.

7.6 Experiment 3: Refinement of the Production Pipeline

Building on the findings of Experiment 2, the third experiment focused on fine-tuning the production process to ensure optimal utilization of all resources. This included:

7.7 Changes Implemented:

- Introducing splitters in the pipeline to balance the distribution of resources, such as copper cables and electronic circuits.




```
Duration of simulation: 132
The amount of resources produced during the simulation time:

Iron ore: 48
Iron plates: 0
Iron plates for Sulfuric Acid: 40
Iron plates for El.circuit: 98
Copper ore: 48
Copper plates: 86
Copper cables: 0
Copper cables for El.circuit: 2
Copper cables for Adv.El.circuit: 36
Electronic circuit queue: 0
Electronic circuit for Adv.El.circuit: 0
Electronic circuit for Processor: 10
Coal: 207
Crude oil for plastic bar: 10600
Petroleum gas for plastic bar: 30
Plastic bars: 100
Advanced electronic circuits created: 3
Water units: 155860
Crude oil for sulfur: 10600
Petroleum gas for sulfur: 30
Sulfur: 6
Sulfuric acid: 690
Processing units: 2
```

7.8 Output Analysis:

The results of experiment 3 demonstrated a more balanced production system. The use of resources reserves from other queues led to an increase in productivity without expanding copper production. The simulation confirmed that the modified production plant can achieve its intended goals using different methods, and the performance of technological equipment reached the highest efficiency in a series of experiments.

8 Conclusion

The conducted simulation experiments and the resulting analysis highlight several important insights into optimizing the processor unit production process in Factorio. One of the most straightforward methods to improve production throughput is increasing the volume of facilities, such as drills, furnaces, and assembly machines. However, while effective, this approach is costly in terms of spatial and resource requirements.

In Factorio, the world is theoretically infinite, but expanding production into larger areas introduces additional challenges, such as increased difficulty in defending against persistent attacks from alien bugs. Expanding production facilities necessitates a corresponding expansion of defensive structures and automation of a more comprehensive defense system. This process is resource-intensive and time-consuming, often outweighing the benefits of increased production capacity.

Conversely, optimizing existing production setups through resource balancing and bottleneck analysis proved to be a more efficient and scalable approach. For instance, introducing additional splitters to balance the flow of resources and better utilizing existing infrastructure led to significant productivity gains. In these experiments, pure expansion increased production output by approximately 50%, while strategic optimizations—such as the introduction of splitters—improved productivity by as much as 75%.

These findings underscore the importance of prioritizing optimization over expansion whenever possible. Analyzing and addressing bottlenecks in the current production system allows for significant gains without the need for excessive resource and space allocation. This principle can extend beyond the simulated environment of Factorio to real-world industrial systems, where efficiency and resource utilization are often critical success factors.

References

1. **The official Factorio Wiki:** https://wiki.factorio.com/
2. **Processing unit in Factorio:** https://wiki.factorio.com/Processing_unit
3. **Petr Peringer. SIMLIB/C++==SIMulation LIBrary for C++**:
   https://www.fit.vut.cz/person/peringer/public/SIMLIB/ , 1991
4. **James L Peterson. Petri nets. ACM Computing Surveys (CSUR), 9(3):223–252, 1977**