3170 Course Project Report

Team 15
Option 1 branch 1 & 2

Abstract

Database Management Systems (DBMS) are widely used in various fields from keeping books ordered in the library to controlling all personal information on the internet. Its ability to manage a complex connection of structured information facilitates information management across a variety of industries, especially for those under sub-contraction mode. Within this project, we create a database for an organization to realize providing a platform for online circuit manufacture orders. In the second half of this report, we analyzed the random generation of package orders, and a Shortest Processing Time Simulation (SPTS) is performed. We also solve the Traveling Salesman Problem considering the travel cost. To enable the scheduling in a complex situation, we further propose an approach using learnable k-means.

1. Introduction to database

1.1. Settings

The basic settings for the organization and manufacture are given as follows:

Consumers can request some plants with some packages. The package refers to the bundle of chips that a consumer requires to finish and each package has an overall time and expense budget itself. Every chip can be of only one chip type. A chip can be either processed by one plant or multiple plants, but it usually requires the cooperation of different machines. Some specific chip types' production requires a few operations to be processed in some partial order. A plant holds machines of multiple types. A machine might be able to process different types of operations, but it can only process one operation at the same time. Multiple plants hold the ability to process an operation, which can be assigned to only one plant in a package. One operation can be processed by one machine at the same time.

In order to more clearly illustrate, an ER diagram is provided here.

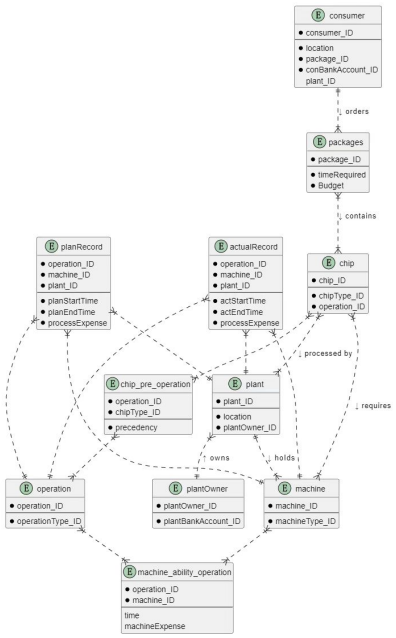

Figure 1. ER diagram

For the database part (branch 1), we have the following settings:

- Geometrical Constraint: Considering the location of consumers and plants, those deal for a consumer and a plant that are too far away from each other can not be made
- Complex Plant: Some plants might hold multiple machine types and can be more flexible in chip processing.
- Centralized Banking System: A centralized banking system is added. A plant can belong to only one plant owner, but one plant owner could have multiple plants. Both the consumer and the plant owner have a bank account. A QR code is provided in the platform for consumers.

Moreover, for the analysis part (branch 2), an extensive setting is shown here:

- Cost for Transportation: If one chip is processed in multiple plants, it additionally requires transportation costs considering the distance of their location. The chips might be transported in batches.

1.2. Functionalities

Our platform provides the following major functionalities:

- Consumers can release their package demand information
- Consumer can appoint some plant for some package manually
- The assignment and the start-time of some operation with some machine could be further set under the constraint of plant appointment
- The processing record in end-time and expense of some operation could be written back once it is successfully finished
- The production information, like the manufacturing capacity of some plants, or the demand changes of some consumers within some period of time can be calculated.

2. Query

We write query to realize the functions about database. To easier understand, the functions can be roughly separated into 3 parts: before, doing, after. The "before" is like to input the information of user after registered, and find the information like location after user logged in. "doing" is working during the choosing and ordering, including functions like screening out the available plant or machine according to the location and types of packages. The "after" is about the work after order, like input information of plan.

Here is an example:

There is a function in web that custom can choose if consider the location, which means if customer and plant are too far away, the plant would not be consider. And the queries for it are chooseProcessNOpianyuan, chooseProcessPianyuan. They are similar in screening out the plant, but one consider the location, one not. Queries can be seen in the file "function", and for the web realization, more details can be seen in the next part.

3. Web page

We designed a simple web page to show our database structure. Given the complexity of the actual situation, we simplified the process and used pre-filled data for the demonstration.

3.1. Registration and login

We designed the user's registration and login page, and used the user's mobile number as the user's ID (primary key). Note that we are not limiting the length of the phone number here, so the user ID in the presentation is three digits.

Figure 2. registration and login page

3.2. Selection and operation

After the login, the operation page is displayed. At the top of this page, some personal information of the user is displayed. The user can select the existing package type in the database (default 5) and view the corresponding description. The package class contains a certain number

of specific chips, and the description also gives the type of operations required to produce the selected package. After selecting the package type, the system will give the user a list of factories to handle the related processes. User can select different factories and see the corresponding situation of the factory, such as factory location, factory owner, factory machine type, etc. Here, we provide a filter function to eliminate factories that are too far away from the user. Note that here we have simplified the constraints by assuming that the processes are independent and can be done separately, so we can only compare the distances between different plants; If the processes are continuous (for example, operation 2 needs to finish operation 1 first), then the cooperation between the factories is required, which we consider to be a traveling salesman problem (TSP), we will talk about TSP in Section 4.

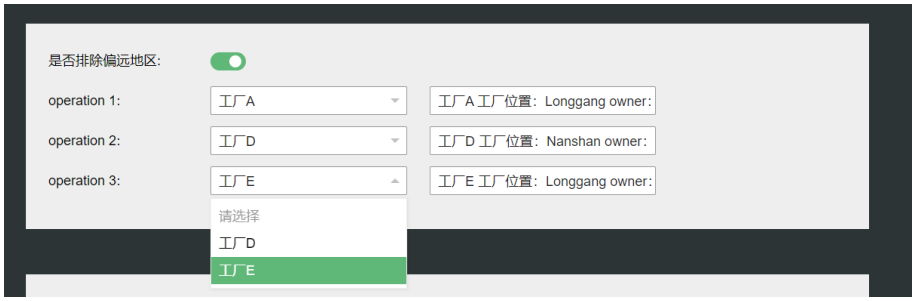

Figure 3. select the factories and view the corresponding situation of the factory

After designating the factory, we can continue to designate the relevant machines and check the processing process, waiting time, cost, etc. We added a feature that automatically matches machines to time optimizations,since We consider this part to be a dynamic programming problem with many constraints. we will discuss this in Section 4.

Figure 4. Designate the relevant machines

3.3. Complete payment

After all the work is completed, click the "Confirm" button, which will jump to the payment page, and scan the QR code to complete the payment.

4. Analysis

4.1. Random Generation

We assume that there are n potential customers and their arrivals follow X_i Poisson(λ), where i indicates customers' id.

It is well known that $\sum_i X_i$ Poisson($n \cdot \lambda$) because of the property of poisson distribution. Now, without loss of generality we further assume there are totally 2 types of chips for clarity

of statements: Chip1 and Chip2. Chip1 requires operation A while Chip2 requires B. Indeed choosing the package first and then invoking arrival is equivalent to first invoking the arrival and then choosing chips in certain probability.

We now show the way to generate poisson arrival in continuous time. Given N_t the number of arrivals during time period t. Y_t the time it takes for one additional arrival assuming someone arrives at time t. By definition, followings hold:

$$(Y_t > y) \equiv (N_t = N_{t+y})$$
$$P(Y_t < y) = 1 - P(Y_t > y)$$
$$P(Y_t < y) = 1 - P(N_{t+y} = N_t)$$
$$P(N_{t+y} = N_t) = P(N_y)$$
$$P(N_{t+y} = N_t) = \frac{\lambda y}{0!} e^{-\lambda y} = e^{-\lambda y}$$

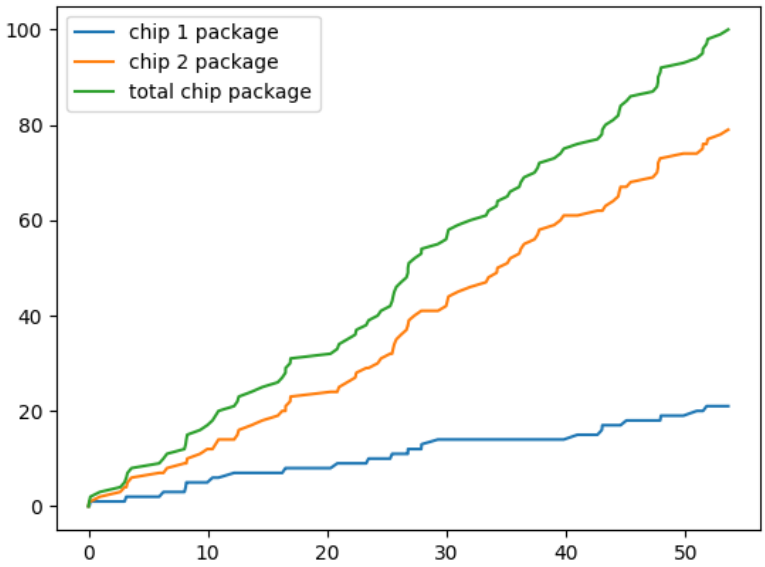

Figure 5. Random arrivals

Which is just the exponential distribution. Note that λ is the rate. In our context, we just need to adjust the rate of different kinds of arrivals. Also, we assume the processing time also follows an exponential distribution. A good property of exponential distribution is that it is memoryless, and thus a time point of an event can "restart" the exponential simulation.

The visualization of chip1 and chip2 arrivals are shown in Figure 5.

4.2. Shortest Processing Time Simulation (SPTS)

We start from the simplest situation. We do not care about the economic cost and the traveling time, but only the processing time. In this case, once there is a request we find the matched machine to process the request package.

Assumption: To simplify the problem, we set four companies: first has 2 machines both for operation A; second has 2 machines both for operation B; third and fourth has 2 machines capable of A and B respectively.

Implementation: We are concerned about the analysis speed. We list several approaches for comparisons:

1. Directly go through plants and machines

2. Keep all machines in a single queue

3. Keep machines in multiple queues for different operations. Maintain pointers from queues to machine info block and from machine info block to queues

The time cost of 1 is $O(m)$, where $m = \sum_{i=1}^{n} m_i$ and m_i is the machine number in plant i

The time cost of 2 is $O(m^{'})$, where $m^{'}$ is the available machine numbers

The time cost of 3 is $O(1)$

Only disadvantage of 3 is that we need to keep the update of queue. For instance, suppose now the machine supports both operation A and B, assigning A to a chip not only removes A from the queue for A but also the queue for B.

The simulation of SPTS is shown in Figure 6. Compared to Figure 5, the increment of remaining packeges is much slower, which is not surprising since the plants now are working.

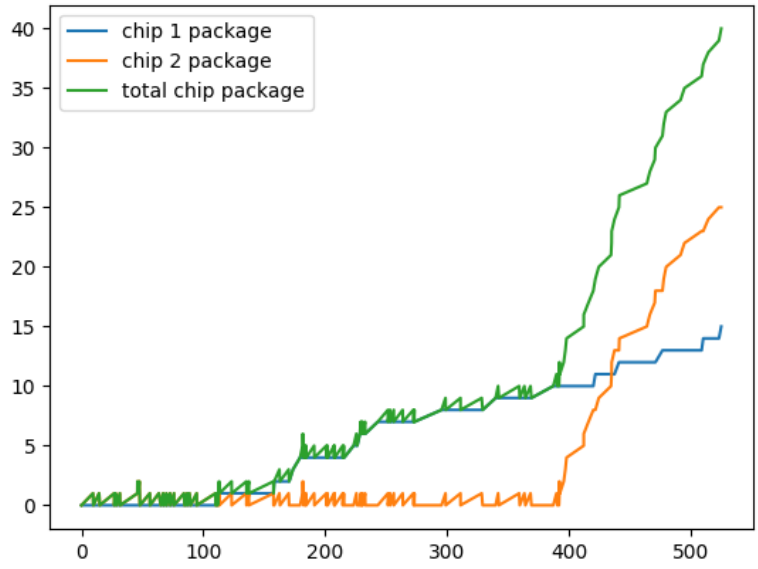

Figure 6. Simulation of SPTS

4.3. Consider Travel Cost: Traveling Salesman Problem

Now if we take the traveling cost (both time and money) into consideration, we need to solve TSP problem. The assumption of the modeling is that there are enough machines in some plants, so that we usually only need to consider the plants that can always offer production services. It is well-known that the exact solution of TSP is NP-complete. However, compared to the cost of traveling, sacrifice of reasonable time of scheduling is acceptable. Now we give some algorithms to solve TSP.Suppose we have N plants

Brute Force: $O(N!)$

Dynamic Programming: $O(N^2 \cdot 2^N)$

Miller Tucker Zemlin formulation: Integer Programming

The demo for solving TSP on a small network is displayed in Figure 7.

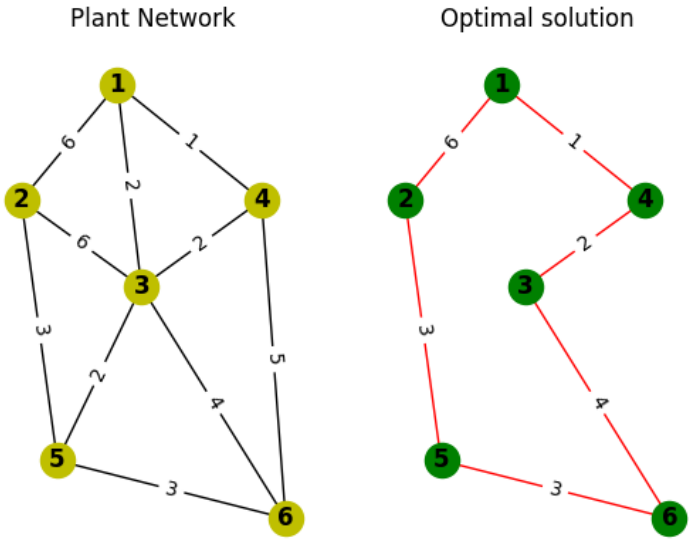

Figure 7. Solving TSP on toy network

4.4. Space Constraint: K-means as heuristics

Given that the TSP is NP-complete, we hope to solve the problem in a more efficient way, especially N is typically very large.

A trivial method is to set a radius around the client and find out surrounding plants. However, here comes the problem:

- What value will the radius take? Too small: no sufficient plant. Too big: not efficient
- What if the circle of many clients overlap?
- Unable to do scheduling in parallel for large number of clients at the same time

To cope with the issue, we try to use k-means with learnable k to do clustering first. The clients are determined to be members of clusters. TSP can be solved in local and thus the complexity is reduced.

We list reference paper for TSP in local:

- Learning the k in k-means.
- Genetic local search algorithms for the traveling salesman problem.

We further illustrate why a learnable k-means is preferable. For traditional k-means, we need to fix the number of centrics, and thus introduce more manual effort. Especially in the decision making problem, we make the decision factor dependent on users, which is inconvenient and may not result in a good clustering. Learnable k-means can automatically divide the global map into local map and one can run the genetic local search for TSP on within the local region and save the traveling cost.

5. Conclusion

In this report, we establish a database for an organization to manage an online platform to register package orders with a series of constraints. We build a simple web page for implementations

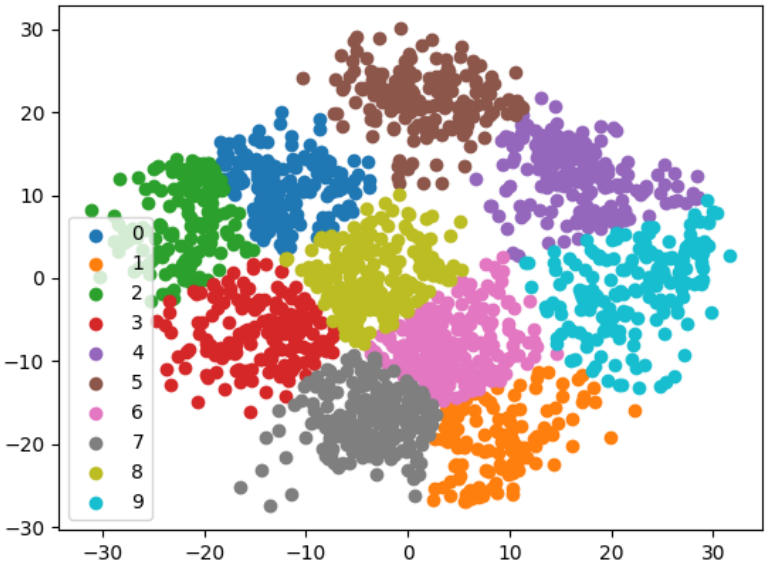

Figure 8. Kmeans demo

of the functions.

Further, we perform analysis and simulation of some specific constraints.

.