**Meeting Scheduling Prototype**

**1. System**

In Fig 1 is detailed all my systems. I will present the core of the system and will skip the usual details so you can understand my intention more easily.

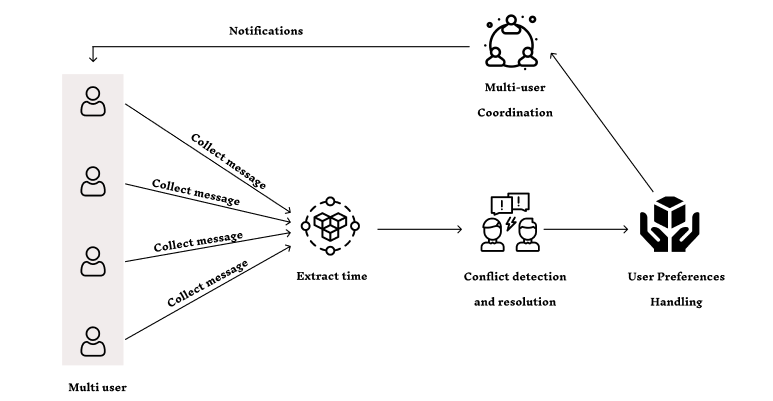


Fig 1: System detail

This system is designed to enhance multi-user parameter scheduling. Initially, user messages are collected from different sources and timings are extracted from these messages. Next, the system detects and resolves conflicts based on the time and preferences of each user. The multi-user coordination process helps to identify conflicts and ensure that all participants can achieve a harmonious schedule. Finally, the system manages user preferences, facilitating conflict resolution and scheduling optimization.

In the **Extraction time** module, there will be an NLU (Natural language understanding ) that takes on the role of understanding the text and extracting all the free time mentioned for each user.

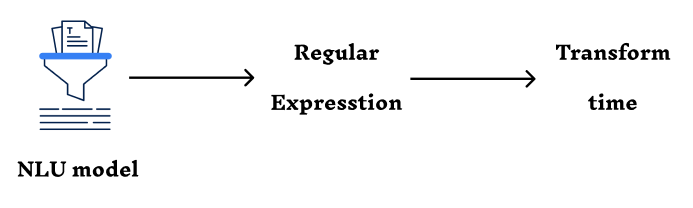


Fig 2: Extraction time module

Then the result is **Regular Expression** to extract time efficiently from various text formats. I used information based rules (datetime) to extract time. Further convert the result to numeric form so that algorithms can be applied.

Why Regular Expression and not NER (Named Entity Recognition)?

In the first time of designing and developing the system, I used NER in my system. But then I thought about whether it is really optimized in terms of cost and performance for the system. Then I thought clearly time has a rule so I decided to find out and I found a set of rules about it. Based on that, I removed NER and replaced it with REGX (**Regular Expression**) because NER is not only really necessary but also very expensive compared to REGX.

The **Conflict detection and resolution** modules will check all available user times and find conflicts. Then I will handle the conflicts and will pick out the appropriate time slots for that user.

The **User Preferences Handling** modules based on user information and text, each user's time priority. These priority times will be used in deciding when to hold the meeting.

**Multi-user Coordination** modules use all the information about user's time and preference i will combine all user’s free time information into a common set combined with preference to give the final meeting time for the user.

The meeting schedule will be announced to all users.

**2. Demo**

About the demo I am using a model QA (Question-Answer) for the demo.

Why QA model?

All the time will be in the message, so based on that feature, I used a QA model to be able to separate the free and busy time in the message and start processing it.

Moreover, currently the APIs of large models are charged over time. After some calculation, I found that working with available resources in the long run will cost less and will avoid dependence on API providers.

**Idea:**

I design systems for companies and work hours are from 9am to 5pm every day, this time can be flexible. After finding out the free and busy times during the week I started using REX for taking time samples to use for algorithms.

Since the meetings will take place during working hours, after getting my templates I will start to find where all the free time is during working time and saturday, sunday in this sample.

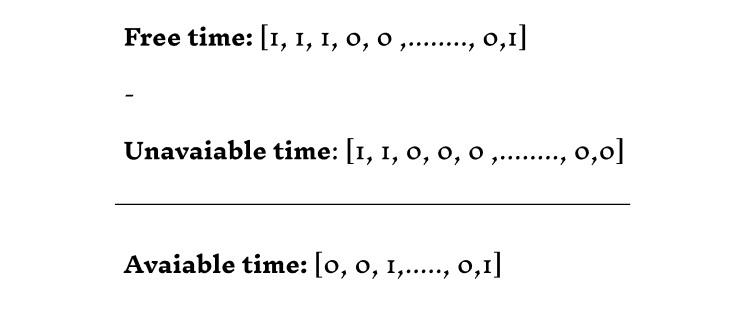


Fig 3: Get available time

Next step i have set up a 1 dimensional array for time with each element in it corresponding to 1 minute. That means from Monday to Sunday corresponds to an array of 3360 elements.

For free time, each element will have 2 values: 0 or 1. If it is 1, it means they are free at that time and if it is 0, it means they are not free. Vice versa for busy times. (Fig 3)

When developing this algorithm declaration I just need to take each element for free unless given the application element and add the condition that if the result from the calculation is -1 then the element will use 0 otherwise the element will use the result.

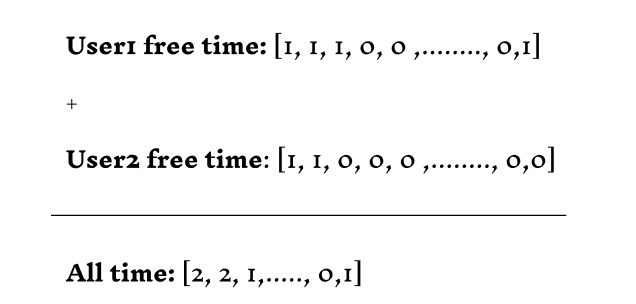


Fig 4: Multi user time.

Given a valid time, I will summation each element with a corresponding element that matches all message senders. This will represent the total number of people free during that time. (Fig 4)

In the end I just need to find the free time in the chain with priority being as many people free as possible.

Finally I just need to find the free time in the string with priority to as many people free as possible and convert it to standard time to display to the user.

Video demo: https://docs.google.com/document/d/1Mxjq45UbiVdFc5HTtasuDxf3tqlnEu2wlX3fi8jaEjs/edit?tab=t.0