國 立 交 通 大 學

網路工程研究所

碩 士 論 文

藥物時程的服用時間彈性和劑量合併之機制

Timing Flexibility and Dose Grouping for Medicine Scheduling

研 究 生：王梅瑛

指導教授：邵家健 博士

中 華 民 國 九 十 八 年 八 月

藥物時程的服用時間彈性和劑量合併之機制

Timing Flexibility and Dose Grouping for Medicine Scheduling

研 究 生：王梅瑛　　Student：Mei-Ying Wang

指導教授：邵家健 博士　　Advisor：Dr. John Kar-Kin Zao

國 立 交 通 大 學

網 路 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Network Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of Master in Computer Science

August 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年八月

藥物時程的服用時間彈性和劑量合併之機制

學生：王梅瑛　　指導教授：邵家健 博士

國立交通大學網路工程研究所　碩士班

摘要

非住院病人的藥物服用是現今醫療保健過程中最容易出錯的一環。多數的服藥錯誤發生在病人同時服用來自不同醫院、診所之處方用藥，和藥局購買的非處方用藥，卻沒有充分了解與遵守用藥指示，尤其年長病人及慢性病患者特別容易發生這種錯誤。本論文中，我們提出 Wedjat，一個在行動手機上使用的藥物提醒與監控工具 (Medication Reminder and Monitor, MRM)以及我們設計的藥物時程劑量分群的機制，目的是減少病人的服藥次數，增加緊急的藥的服用機會，同時增加劑量服用時間的彈性，我們利用多維評估因素決策方法來決定最佳的分群組合。Wedjat 有三個主要的特色: (1)遵守藥物指示與處方籤，計劃適當的服藥時程，同時避開潛在的藥物-藥物/藥物-食物交互作用；(2)控制服藥劑量，保持在適當的安全劑量範圍；(3)將時程接近的藥，合併在一起服用，減少繁複的提醒。Wedjat 會維護使用者所有的用藥記錄，和家中主機的資料庫進行同步或上傳至遠端個人健康記錄伺服器(Personal Health Record Server, PHR)，做為往後醫療之用。

Timing Flexibility and Dose Grouping for Medicine Scheduling

Student: Mei-Ying Wang Advisor: Dr. John Kar-Kin Zao

Institute of Network Engineering

National Chiao Tung University

ABSTRACT

Out-patient medication administration has been the most error-prone process in contemporary health care. Most medication administration error was made when patients take various medicines from different sources at one time and with little or no directions, such as prescriptions from different doctors, over-the counter medicines from several drug stores and nutritional supplements. Especially, Elderly or chronic ailment patients are likely to make these mistakes.

In this paper, we present Wedjat, a mobile phone based medication reminder and monitor to prevent these medication administration errors. We design a mechanism of dose grouping to decrease the number of in-take times and assign each in-take to a sufficiently long time slot and increase chances of taking emergency doses. Wedjat remind patient to take correct doses in right time and note down the in-take record for future medical care use. Three significant features of Wedjat are: (1) plan a proper intake schedule obey all intake medication directions and prescriptions and avoid all latent medicine-food/ medicine-medicine interactions; (2)

control in-take dosage rate in safe range; (3) group in-take doses which in-take time are close to each others to prevent complicated remind. Wedjat maintains patient's intake records and synchronize with local host database or upload to remote Personal Health Record Server.

致 謝

首先感謝邵家健老師兩年來的指導，無論是在學術研究或做人處事方面，老師都給予我莫大的幫助與啟發，真的很謝謝老師。

此外，我想感謝實驗室的各位同學，博政，嘉錡，星閔，勝焜，彥霖，兩年來，有他們互相支持陪伴，讓我實驗室的生活變得更加愉快，在課業問題上遇到問題，他們也都會非常不遺餘力的給與我協助與建議。實驗室的學弟妹們，子晉，家妤，志明，俊偉，鈞凱，大家一起吃飯聊天，互相分享心情，讓實驗室的生活增添不少樂趣，也謝謝俊瑋和子晉，常常幫我解決許多技術上的問題。

最後，我要謝謝所有可愛的親朋好友，還有我的爸爸媽媽，他們是默默支持我鼓勵我的主要力量，沒有你們，我不可能完成這份論文。感謝我的男朋友，學增，雖然這一年他不在我身邊，不過總是在我遇到挫折，沮喪難過的時候，給我加油打氣，幫我信心喊話，謝謝你的支持。

要感謝的人太多了，真的謝謝所有鼓勵過我的你(妳)。

目 錄

圖目錄

表目錄

第 1 章　綜論

1.1　問題陳述

根據一項指標性的醫療錯誤研究顯示[1]，用藥錯誤 (medication errors, ME)和藥物[1]衝突反應 (adverse drug reactions, ADR)是最常見的醫療錯誤。儘管藥物誤用可能發生在醫療過程中的個個環節，像是藥品的採購，處方籤的開立，藥師配藥、給藥，病人服用藥物等等，然而，最容易發生錯誤的是服藥階段。

多數的服藥錯誤發生在病人服用來自不同醫院、診所之處方用藥，和藥局購買的非處方用藥，卻沒有充分了解與遵守用藥指示，尤其年長病人及慢性病患者特別容易發生這樣的錯誤。常見的服藥錯誤有: (1) 忙碌的生活方式下不規律的服藥，導致劑量過量或不足；(2) 服藥的種類繁多，導致無法遵守複雜的服藥時程；(3)未經調和的多張處方籤，造成藥物之間的交互作用；(4)缺乏適當服藥的知識；(5)有服藥上的疑問時，缺乏向醫藥人員的諮詢；(6)缺乏監控、追蹤服藥的機制。近來，遠距醫療和遠距監控技術，已經被認為是有效益提升非住院病人醫療照護品質的方法[2,3,4,5]。我們提出一套在手機上使用的藥物提醒工具，並且根據已發展的排程演算法和排程規格做改進，本論文將針對藥物時程的劑量時間彈性與劑量分群提出一個改善機制，希望能有效減少病人的服藥次數，增加緊急的藥的服用機會，同時增加劑量服用時間的彈性。

[1] 本論文中的藥物泛指處方用藥、非處方用藥以及營養保健補給品

1.2 研究方法

在本論文中，我們將提出一套藥物時程劑量分群的機制以及 Wedjat 的設計與實作。劑量分群的機制利用多維評估因素決定方法，並根據藥物的重要性，階層化評估因素之間的相對比重，來找到滿足多維考量下的最佳的劑量分群組合，在病人錯過分群劑量時，也能依照多維評估找到修改後的分群組合。

Wedjat 有下列三個主要的功能:

1) 發出服藥提醒：Wedjat 會在病人需服藥的前 15 分鐘，發出用藥提醒，它會反覆提醒直到病人確認服藥或取消。我們用一個已發展的排程演算法 [2.2.2.1 小節] 來計算服藥的時程，然後我們根據這個時程，做一些延伸改進，主要的目的是增進使用者服藥的彈性與方便性，我們將在第三章和第四章說明我們設計的改進方法。Wedjat 應用程式需安裝在行動裝置上，服藥提醒的功能將與行動裝置內建的應用程式行事曆 (Calendar) 結合。
2) 提供藥物辨別及服用的資訊：Wedjat 有一個內建的資料庫，紀錄藥物的重要資訊 (包括藥品的照片，用藥指示，和注意事項等等)，以及和病人相關的診所、醫院的資訊。部分的資訊會在服藥提醒通知中出現，完整的資訊皆可從 Wedjat 提供的介面進行查詢。
3) 維護使用者的用藥紀錄：Wedjat 會紀錄病人所有的用藥情況，包含什麼時間吃了什麼藥，那些藥錯過沒有服用。 這些紀錄可以與家中主機上的資料庫進行同步，或者上傳至遠端個人健康紀錄系統。

1.3 論文大綱

接下來的文章將分成六個章節：第二章說明基本的知識背景，包括排程的用藥規格，排程演算法 (OMAT & ODAT)，與多維決策決定法 (MCDM)。第三章說明我們對服藥時程所做的延伸，包括我們計算彈性時程的方法，以及改善劑量彈性時間的 OMAT 排程。第四章說明我們如何將劑量的時程合併之方法。第五章介紹我們實作的方法，包括系統的設計與操作原理與介面的設計等。最後一章說明我們的貢獻以及未來可行的研究方向。

第 2 章　背景知識

2.1　藥物排程規格書

藥物排程規格書 (Medication Scheduling Specification, MSS [6]) 包含用藥指示的資訊 (medicine direction)，明確說明用藥的規則與限制，是一個機器可讀的格式，也是排程演算法的輸入資料的格式。MSS 主要包含兩個部分，用藥參數 (Dosage Parameters, DP) 和特殊參數 (Special Instructions, SI)。

2.1.1　用藥參數

DP 包含了藥物本身的資訊和服用規則的參數，如表格 1 所示。包括藥物的名稱 M，藥物的最小單位劑量 G，療程長度以及劑量大小和服用時間間隔等限制。最小單位劑量 G 通常以 tablet/ capsule 等為單位，其它與劑量有關的參數皆以 G 為單位來表示，ex, k G 代表 k (tablet/capsule/..)，而本文中所提及的時間參數，皆以一小時為單位。

服用的劑量範圍[dmin, dmax]，也就是每次服用的劑量必須介於最小劑量和最大劑量之間。服用時間間隔為[smin, smax]，也就是任兩個連續劑量要保持在最小時間間隔和最大時間間隔之間。以 Advil [7]為例，用藥指示說 “take 1 gel caplet every 4 to 6 hours. If pain or fever does not respond to 1 caplet, 2 caplets may be used.”，所以相對應的時間間隔參數為 [4, 6]，劑量範圍參數為[1, 2]。

在某一長度的時間內，可服用的最小與最大劑量限制為 (L, P)和 (B, R)。舉例來說，“The smallest effective dose should be used and do not exceed 6 gel caplets in 24 hours unless directed by a doctor.” ，對應到 MSS 裡的參數，最低劑量限制 (L, P)的值為 (0, 24)，最高劑量限制 (B, R)的值為 (6, 24)。

絕對時間間隔參數[Smin, Smax]和絕對劑量範圍參數[Dmin, Dmax]提供較寬鬆的時間和劑量限制。一般而言，[smin, smax]包含於[Smin, Smax]，[dmin, dmax]

包含於[Dmin, Dmax]。舉例來說，Fosamax [7]的用藥指示內容提到“The usual dose of Fosamax is 10 milligram once a day or 70 milligram once a week.”相對應的一般時間間隔參數為[24, 24]，一般劑量範圍為[1, 1]，絕對時間間隔參數為[24, 168]，絕對劑量範圍為[1, 7]。排程時可利用絕對參數限制得到較大的彈性。

2.1.2 特殊參數

MSS 第二部分的參數是特殊參數 (Special Instructions, SI)。SI 參數說明該藥物 M 與其他藥物或食物之間的交互作用關係。Interferer <List>描述 M 與所有其他有衝突關係的藥物(食物)N 的服用關係，舉例來說，“The smallest effective dose should be used and do not exceed 6 gel caplets in 24 hours unless directed by a doctor.”，意指這兩類藥物在服用時至少要間隔兩小時以上，假設 M 為 Warfarin，則對應到 Interferer 的參數的結果，N 為 Crestor ，minfrinterferer 和 mintointerferer 的值皆為 2。一般來說，這兩組參數的值不一定相等。

SI 的另一部分是 DosageParameters <List>，用來反映當 M 被其他藥物影響時，所對應的用藥參數的改變，例如劑量的減小或服用頻次的改變。

Dosage Parameters	
M	Name of the medication
G	Granularity
[Tmin, Tmax]	Minimum and maximum duration over which the medication is administered
[dmin, dmax]	Nominal minimum and maximum dose size
[smin, smax]	Nominal minimum and maximum separations between (consecutive) doses
[Dmin, Dmax]	Absolute minimum and maximum dose size
[Smin, Smax]	Absolute minimum and maximum separations
(B, R)	Maximum intake over a specified time interval given by budget *B* and replenishment delay *R*
(L, P)	Minimum intake over a specified time interval given by lower bound *L* and interval length *P*
Special Instructions	
Interferer <List>	Interacting medications interferes *N* with M
DosageParameters <List>	Required changes of DosageParameters when some interferer affects M

表格 1: 藥物排程規格書

Interferer	
N	Name of interferer medication
minfrinterferer	Minimum separation from interferer N to medicine M
mintointerferer	Minimum separation from medicine M to interferer N

表格 2: 衝突參數

2.2 藥物排程演算法

目前有兩個已發展的藥物排程演算法，One-Medicine-at-A-Time (OMAT) 和 One-Dose-at-A-time (ODAT) [8]，它們是根據即時系統排程理論[9]所設計出來的演算法，這一章將會介紹這兩個演算法它們所運用的模型，以及它們的主要概念和用途。

2.2.1 排程模組

2.2.1.1 資源模組

OMAT 和 ODAT 運用即時系統排程理論的資源模組 (Resource Model) 的概念。它在排程時分配給每一藥物一個虛擬處理器 (*virtual processor PM*) 和一個虛擬資源 (*virtual resource RM*) 來處理排程時的相互時間關係。*PM* 負責處理同一藥物的每劑服用時間的間隔關係 (intra-dose separations)，*RM* 負責處理有交互作用的藥物(或食物)之間的服用時間間隔關係 (inter-dose separations)。當我們在計算服藥時程時，我們將每一藥物 (medicine) 視為一個任務 (task) ，每一劑 (dose) 的服用視為一個在 *PM* 上執行的工作 (job)，一連串的工作組合成一個任務。一個工作的起始點 (start time) 代表使用者必須服藥的時間點 (in-take time)。一個藥物的時程就是一個時間表，每一個時間代表每一劑藥物必須被服用的時間點。

2.2.1.2 資源分配規則

排程演算法利用 *PM* 來處理連續劑量之間的正確時間間隔。它將每一劑藥物視為一個在 *PM* 上執行的非搶占式工作 (non-preemptable job[1])，每一個工作的執行時間 (execution time)為最小時間間隔 `smin`，也就是每一個工作會占用 *PM* 最小時間間隔長度的時間，*PM* 同一時間只能執行一個工作。另外，它將最大時間間隔長度視為兩個連續工作之間的相對期限時間 (relative-deadline)，以確保連續劑量的服用時間不超出最大時間間隔。為了避免藥物-藥物的衝突或藥物-食物的衝突，排程演算法利用 *RM* 來處理衝突藥物(食物)之間的服用時間間隔關係。任一個藥物被排程時，會占用所有與它有衝突關係的藥物的虛擬資源 *RM*，從該工作開始的時間點往前占用 `minfrinterferer` 時間，往後占用 `mintointerferer` 時間。換句話說，它利用占用別人的資源來擋住與它有衝突關係的藥物，也就是任何一個工作只有在 *PM* 和 *RM* 沒有被占用時，才能開始執行。

2.2.1.3 優先順序方案

OMAT 和 ODAT 根據藥物的優先順序 (priority) 來做為排程的順序。它利用幾種優先順序的決定方法，分配給每種藥物優先順序，演算法在執行時會從最高優先順序的藥物排到最低優先順序的藥物。優先順序決定方法有下列五種：

1) MVF (most-victimized-first)：它根據最糟情況的阻斷時間 (worst case blocking time) 來決定優先順序，也就是在最糟情況下，RM 可能被占用的時間長度，時間長度越長，優先順序越高。
2) MIF (most-interactions-first)：它根據衝突藥物的數量來決定優先順序，也就是衝突藥物越多的藥物，優先順序越高。
3) SSDF (shortest-separation-difference-first)：它根據時間間隔的差距長度來決定優先順序，也就是最小時間間隔跟最大時間間隔的差，相差越短的，優先順序越高。

[1] 一個非搶占式工作是一種從執行的開始到結束都不容許被中斷的工作
(A job is non-preemptable if it must be executed from start to completion without interruption)

4) RM (rate-monotonic)：它根據藥物服用的週期來決定優先順序，週期越短的，優先順序越高。

5) EDF (earliest deadline first)：這是一種動態決定優先順序的方法，它會根據現有工作的期限時間來決定不同劑量的優先順序，期限時間越早的，優先順序越高。

2.2.2 排程演算法

這一小節介紹 OMAT 與 ODAT 的演算法概念，表格 3 和表格 4 分別為 OMAT 和 ODAT 的虛擬碼，它們使用一個布林植 `feasibleSchedule` 來表示排程器是否排出一個合理的服藥時程，當 `feasibleSchedule` 的值為真時，`MIS` 陣列的內容即為正確排程的結果，也就是所有藥的服藥時程。

2.2.2.1 一次一藥法 (One-Medicine-at-A-time, OMAT)

OMAT 排程演算法會根據藥物的優先順序，一次排一種藥，依次排完所有藥的服藥時程。在排程時，OMAT 會排出每種藥在療程期裡的所有劑量 (dose) 的服用時程。它利用 Resource Model 來確保所有劑量之間的安全時間間隔。OMAT 排程法會指定給每種藥一個固定的優先順序，所以 EDF 優先順序決定法在 OMAT 裡不適用。

OMAT 試著排出所有藥的時程，而且在遵守用藥規則和限制上，表現的比 ODAT 更好，但是 OMAT 適合傾向準時服藥的使用者，因為當使用者延遲或遺忘吃藥時，整個時程就必須重排。

Basic OMAT 在排程時會盡可能排出緊密的時程，讓每劑 (dose) 的服用時程越接近越好，這也導致排程失敗的機率較高，因為當有藥物衝突時，較高優先順序的藥可能把時間都占滿，以致低優先順序的藥很難成功排程。為解決此問題所發展出的另一版本的 OMAT，稱為 advanced OMAT，它在排程時會將每劑的服用時程盡可能的安排在靠近期限時間 (respective deadline)，也就是它會盡可能排出時間鬆散的時程，大幅提高了排程成功的機率。

2.2.2.2 一次一劑量法 (One-Dose-at-A-time, ODAT)

ODAT 排程演算法每次只排出下一劑的服用時程，它會根據 EDF 動態優先順序決定法來決定下一個最急迫的藥物來排程。當使用者服完這個劑量後，ODAT 會再決定下一劑的服用時程，所以我們無法預先知道往後的時程。ODAT 在遵守藥物的規則和限制上表現的較差，但是 ODAT 在處理使用者即時的服藥行為，表現的比 OMAT 好，所以 ODAT 適合傾向於延遲吃藥的使用者。

```
INPUT:
    ListMedicationScheduleSpec MSS[number_medicines]
    PrioritySchemes PS
OUTPUT:
    Boolean feasibleSchedule
    List MedicineIntakeSchedule MIS[number_medicines]
PROCEDURE:
    FOR every MSS[i]{
        CREATE JobModel, ResourceModel
        CREATE integer latestStartTime }
    ASSIGN priority to every MSS[i]
        based on PrioritySchemes PS
    SET feasibleSchedule to TRUE
    FOR every MSS[i] according to descending priority{
        SET latestStartTime of MSS[i] to 0
        SET ResourceModel.feasible of MSS[i] to TRUE
        WHILE ResourceModel.feasible of MSS[i] is TRUE {
            FOR i from JobModel.release_time to
                        JobModel.deadline of MSS[i] {
                IF ResourceModel.processor[i]==0 and
                        ResourceModel.resource[i]==0 {
                    SET Tavail to I THEN BREAK }}
            IF Tavail is found
                IF Tavail>=MSS[i].TheraphyDuration THEN BREAK
                APPEND Tavail to ResourseModel.Schedule
                SET latestStartTime of MSS[i] to Tavail
                SET JobModel.release_time to (Tavail+ nsmin)
                SET JobModel.deadline to
                        (JobModel.release_time + asmax)
                FOR k from Tavail to (Tavail + nsmin)
                    SET ResourceModel.process[k] of MSS[i] to 1
                FOR every interferer MSS[i].IP[j]{
                    FOR k from {
                    (Tavail – MSS[i].IP[j].minFrInterferer) to
                    (Tavail + MSS[i].IP[j].minToInterferer)
                        SET ResourceModel.resource[k] of
                                MSS[MSS[i].IP[j].med_id] to 1 }}
            ELSE
                SET ResourceModel.feasible of MSS[i] to FALSE
                SET feasibleSchedule to FALSE }
        IF feasibleSchedule is TRUE {
            SET MIS[i] to ResourceModel.Schedule of MSS[i]}
    RETURN feasibleSchedule and MIS[]
END PROCEDURE
```

表格 3: OMAT 虛擬碼

```
INPUT:
    List MedicationScheduleSpec MSS[number_medicines]
    PrioritySchemes PS
OUTPUT:
    Boolean feasibleSchedule
    List MedicineIntakeSchedule MIS[number_medicines]
PROCEDURE:
    FOR every MSS[i]
        CREATE JobModel, ResourceModel
    SET numberMedicines to the length of MSS[] array
    CREATE integer array
        ReleaseTimeCurrentJobs[numberMedicines]
    SET ScheduleComplete to 0
    SET feasibleSchedule to TRUE
    WHILE (feasibleSchedule is TRUE) and
            (scheduleComplete < numberMedicines){
        FIND MSS[i] with earliest time in
            ReleaseTimeCurrentJobs[] and highest priority
        FOR k from JobModel.release_time
                    to JobModel.deadline of MSS[i]{
            IF ResourceModel.processor[k]==0
                and ResourceModel.resource[k]==0{
                SET Tavail to k THEN BREAK }}
        IF Tavail is found
            IF Tavail >= MSS[i].TheraphyDuration {
                INCREMENT scheduleComplete THEN BREAK }
            APPEND Tavail to ResourseModel.Schedule
            APPEND ResourceModel.Schedule to MIS[i]
            SET JobModel.release_time to (Tavail+nsmin)
            SET JobModel.deadline to
                    (JobModel.release_time + asmax)
            SET ReleaseTimeCurrentJobs[i] to(Tavail+nsmin)
            FOR k from Tavail to (Tavail + nsmin)
                SET ResourceModel.process[k] of MSS[i] to 1
            FOR every interferer MSS[i].IP[j] {
                FOR k from
                (Tavail - MSS[i].IP[j].minFrInterferer) to
                (Tavail + MSS[i].IP[j].minToInterferer) {
                    SET ResourceModel.resource[k] of
                            MSS[MSS[i].IP[j].med_id] to 1 }}
        ELSE
            SET feasibleSchedule to FALSE }
    RETURN feasibleSchedule and MIS[]

END PROCEDURE
```

表格 4: ODAT 虛擬碼

2.3 多維評估因素決策方法

多維評估因素決策方法 (Multi-Criteria Decision Making Methods, MCDM [10]) 是一種幫助決策者在考慮多方面且互相矛盾的評估下，做出較佳決策的方法。它的目標在於用一種透明化的方式，突顯這些互相矛盾的評估，找出一個較佳的妥協方案。

2.3.1 決策矩陣的定義

多維度決策問題可以用一個簡單的數學矩陣表示。如圖 1 所示，一個決策矩陣 (Decision Matrix) 是一個 (m × n) 的矩陣，橫軸代表所有的評估因素 (或稱為目標)，縱軸代表所有可能的選擇。a_{ij} 的數值代表在 C_j 這個維度的評估下，A_i 這個決定的效能表現好壞 (i = 1…m, j = 1…n)。假設決策者已經決定所有評估因素之間的相對重要性，也就是相對的比重 (relative weight)，W_j 即代表 C_j 所占的比重。

多維度決策矩陣的定義如下:

$A = \{ A_i, \text{for } i = 1, 2, 3, \ldots, n\}$

$C = \{C_j, \text{for } j = 1, 2, 3, \ldots, m\}$

A 和 C 皆是一個有限的集合 (finite set)，A 包含所有可能選擇的決定，C 包含所有決策者考量的因素，也就是期望的目標。多維度決策方法會考量所有相關的評估因素，找出一個最佳的選擇 A^*。

Alts.	C_1 (W_1)	C_2 (W_2)	… …	C_n (W_n)
A_1	a_{11}	a_{12}	…	a_{1n}
A_2	a_{21}	a_{22}	…	a_{2n}
.			…	
.			…	
A_m	a_{m1}	a_{m2}	…	a_{mn}

圖 1: 典型的決策矩陣

2.3.2 比重加權法

比重加權法 (Weighted Product Model Method, WPM) 利用相對數值比較來算出兩種選擇 (A_k 和 A_L) 在多維度考量下的優劣。它的計算方式如下列公式:

$$R(A_K/A_L) = \prod_{j=1}^{n} (a_{Kj}/a_{Lj})^{w_j} \quad (1)$$

比重加權法計算兩種選擇在每一評估因素的分數的比值，然後根據每一評估因素所占的比重，做指數次方，再將每一維度所算出來的值相乘。結果大於 1 的話，表示 A_k 這個選擇比 A_L 好，反之如果數值小於 1，表示 A_L 比 A_k 好。在 m 種選擇之下要找出最佳選擇，最多需要比較 C(m, 2) 種組合，也就是任兩種可能的組合。

舉例來說，如果有一個多維度決策矩陣的數值如圖 2 所示:

Alts.	C_1 (0.20)	C_2 (0.15)	C_3 (0.35)	C_4 (0.30)
A_1	25	20	10	30
A_2	10	30	20	30
A_3	30	10	30	20

圖 2: 決策矩陣範例

利用 WPM 的方法計算，A_1 和 A_2 的比較值為:

$R(A_1/A_2) = (25/10)^{0.20} \times (20/30)^{0.15} \times (10/20)^{0.35} \times (30/30)^{0.30} = 0.8867 < 1$

所以在這種計算方法下，A_2 比 A_1 好。

2.3.3 修正後之層級分析法

修正後之層級分析法 (Revised Analytic Hierarchy Process, R_AHP) 利用相對重要性的概念，來比較不同選擇之間的優劣。它會將原本的多維決策矩陣的數值，轉換成相對的比較值。舉例來說，假設原本的矩陣為圖 2，在 R_AHP 方法中，會將這個矩陣轉換成相對重要性比較值的矩陣，轉換後的結果如圖 3 所示:

Alts.	C_1 (0.20)	C_2 (0.15)	C_3 (0.35)	C_4 (0.30)
A_1	25/30	20/30	10/30	30/30
A_2	10/30	30/30	20/30	30/30
A_3	30/30	10/30	30/30	20/30

圖 3: 修正後之層級分析法決策矩陣範例

它將每個維度的數值，除以該維度裡的數值的最大值，然後再依照下列公式，計算每種選擇的分數，分數最高的是此方法中的最佳選擇。

$$A_{R\_AHP-score} = \max_i \sum_{j=1}^{n} a_{ij} w_j \quad (2)$$

所以 A_1 的分數為:

$A_{1,\ R\_AHP\text{-}score} = (25/30)\times 0.20 + (20/30)\times 0.15 + (10/30)\times 0.35 + (30/30)\times 0.30 = 0.683$

同樣的方式，我們可以計算 A_2 和 A_3 的分數:

$A_{2,\ R\_AHP\text{-}score} = 0.749$

$A_{3,\ R\_AHP\text{-}score} = 0.8$

所以這三種選擇的好壞順序為:

$A_3 > A_2 > A_1$

第 3 章　藥物時程之服用延伸一–服藥時間彈性

3.1　原理

我們利用 OMAT 演算法 [2.2.2.1] 的排程結果做服藥時間的延伸。我們計算 OMAT 裡每一劑服用時間的可容忍彈性時間範圍，依據此彈性時間範圍，我們可將時程接近的劑量合併。由於劑量合併時考量許多因子，所以我們採用多維決策決定法，來找出最佳的合併組合，下一節將詳細闡述我們設計的方法。

3.2　操作機制

3.2.1　修改之用藥排程規格書

我們把原本的 MSS 做小幅度的修改，修改後的 MSS 可分成三個部分。第一部分是與處方籤內容有關的參數 (Prescription Parameters)，包含處方籤內有的資訊，像是藥物的名稱、劑量、療程等等；第二部分是劑量參數 (Dosage Parameters)，與原本的 MSS 類似，不同的地方是我們精簡了某些參數，我們將原本的一般服用時間間隔 [nsmin, nsmax] 精簡為服藥的週期 period，由於一般處方用藥會有一定的給藥頻次，例如，一天三次，我們可以從這個服藥頻次轉換成服藥的週期，對於一些非處方用藥規則中的一般服藥時間間隔，我們則可以取這個範圍內的某個值為服藥週期，例如，nsmin 和 nsmax 的中間值。第三部分是藥物的衝突參數 (Interaction Parameters, IP)，每一個衝突藥物會用一組衝突參數來表示。我們精簡 Special Instruction 裡面的 change Dosage Parameters 並將裡面的衝突參數獨立出來，我們將動態改變 MSS 的資料處理方法視為實作上的考量，所以暫時省略 change Dosage Parameters。

Prescription Parameters (PP)	
Medicine Identifier	*M*
Medicine Dose	*g*
Medicine Form	Capsule/Tablet/…
Medicine Amount	*n*
Therapy Duration	*T*
Dosage Parameters (DP)	
Nominal Frequency Separations	`period`
Max. In-take Amount *B* over Interval *R*	`(B, R)`
Min. Intake Amount *L* over Interval *P*	`(L, P)`
Absolute Min. & Max. Dose	`[Dmin, Dmax]`
Absolute Min. & Max. Separations	`[asmin, asmax]`
Interaction Parameters (IP) <List>	
Interferer Identifier	*N*
Min. Separation from M to N	minToInterferer
Min. Separation from N to M	minFrInterferer

表格 5: 修改後的藥物排程規格書

3.2.2 計算藥物時程之服用時間彈性

3.2.2.1 無衝突藥物的服用時間彈性

沒有衝突的藥物排程中，每種藥物的時程是獨立的、不互相影響的，所以我們可以就單一藥物 OMAT 排程的結果來計算每劑的可服用彈性時間範圍 (flexibility)。

假設藥物 A 總共需服用 n 劑 (A_1….A_n)，每劑的時程點依序為 t(A_1)…t(A_n)。根據藥物服用規則限制，任何連續劑量間，須滿足時間間隔限制，也就是任一劑量 A_i 須與前後劑量 (A_{i-1} 與 A_{i+1}) 保持在適當的時間間隔範圍 `[asmin, asmax]`，我們定義 A_i 的可服用彈性時間範圍為 I(A_i)，時程點的左半段區間稱為*可提早的時間範圍 (allowable advance)*，時程點的右半段區間稱為*可延遲的時間範圍 (allowable tardiness)*。

A_i 在滿足與前一劑量 A_{i-1} 的時間限制下，彈性時間的區間範圍如下：

$$I^{i-1}(A_i) = \left[t(A_{i-1}) + \text{asmin}, \quad t(A_{i-1}) + \text{asmax} \right] \quad (3)$$

A_i 在滿足與下一劑量 A_{i+1} 的時間限制下，彈性時間的區間範圍如下：

$$I^{i+1}(A_i) = \left[t(A_{i+1}) - \text{asmax}, \quad t(A_{i+1}) - \text{asmin} \right] \quad (4)$$

所以 A_i 在滿足兩者的限制下，可服用的時間範圍為(3)與(4)的交集：

$$I(A_i) = I^{i-1}(A_i) \cap I^{i+1}(A_i) \quad (5)$$

舉例來說，假設藥物 A 的服用頻次為每八小時一次，時間間隔限制為 [6, 10]，OMAT 的排程結果如圖 4 所示，我們可以看到，A_2 在滿足與 A_1 的間隔限制下，可服用的範圍為 [6, 10] (藍綠色區間)，A_2 在滿足與 A_3 的間隔限制下，可服用的區間為 [6, 10] (橘黃色區間)，所以由公式(5)我們可以推出 A_2 的*可服用時間範圍*為 $I(A_2)$ = [6, 10]，這段時間範圍可分成，*可提早的時間範圍* [6, 8] (圖 4 的(1))，*可延遲的時間範圍* [8, 10] (圖 4 的(2))。我們可以用同樣的方式計算每一劑的可服用時間範圍。

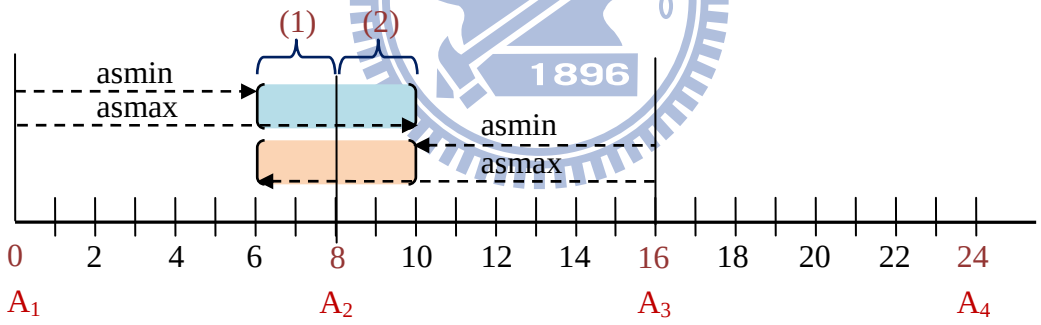


圖 4: 單一藥物服用時間彈性範例

3.2.2.2 衝突藥物的服用時間彈性

當藥物之間有交互作用時，可服用時間範圍的計算方式與前一小節的方法相似，唯一不同的是要考量衝突藥物之間的間隔限制。我們可以利用 Resource Model 來計算有衝突之劑量的彈性時程。此時要考量的限制因素有兩點：

1) 同一藥物的連續劑量之間的間隔限制

2) 衝突藥物之間的間隔限制

滿足第一條限制下的可服用彈性時間範圍可利用前述方法來計算 (公式(5))。滿足第二條限制下的可服用時間範圍，可利用藥物 M 的虛擬資源 (Virtual Resource, []) 被占用的情況來計算。計算方式為從該劑量的時程點往前和往後計算 Virtual Resource 沒有被占用的連續時間長度，我們稱這段區間為 $I^{inter}(M_i)$，即為 M_i 劑量在滿足條件 2 下的可服用區間。

因此，任何一個有衝突劑量 M_i 的可服用時間範圍的計算方式如下：

$$I(M_i) = I^{i-1}(M_i) \cap I^{i+1}(M_i) \cap I^{inter}(M_i) \quad (6)$$

$$I(A_i) = I^{i-1}(A_i) \cap I^{i+1}(A_i) \cap I^{interN1}(A_i) \cap \dots \cap I^{interNm}(A_i)$$

$$I^{interNj}(A_i) = [t_p^{Ai}(N_j) + \text{minToInter}(N_j), t_n^{Ai}(N_j) + \text{minFrInter}(N_j)]$$

舉例來說，假設使用者同時服用 Crestor 和 Warfarin 兩種藥，它們之間的關係如圖 5 所示，Crestor 的時間間隔限制為 [10, 14]，Warfarin 的時間間隔限制為 [4, 6]，它們之間的衝突間隔限制皆為兩小時。

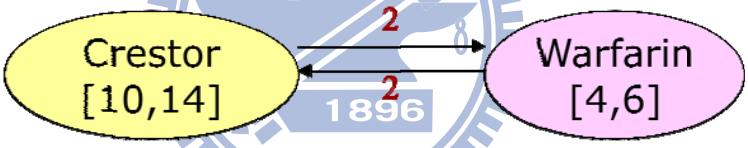


圖 5: 藥物衝突關係之圖例

假設圖 6 為 Crestor 和 Warfarin 的排程結果，P(C)和 R(C)分別代表 Crestor 的虛擬處理器 PM 和虛擬資源 RM，P(W)和 R(W)為 Warfarin 的 PM 和 RM，Crestor 的服用時程為 (0, 12, 24…)，Warfarin 的服用時程為 (4, 9, 15…)。以 C_2 為例，可服用區間 $I(C_2)$為：

$I^1(C_2)$ = [0＋10, 0＋14] = [10, 14]

$I^3(C_2)$ = [24－14, 24－10] = [10, 14]

$I^{inter}(C_2)$ = [11, 13]

$I(C_2)$ = $I^1(C_2) \cap I^3(C_2) \cap I^{inter}(C_2)$ = [11, 13]

同樣的方法，我們可以算出每劑的可服用時間範圍。

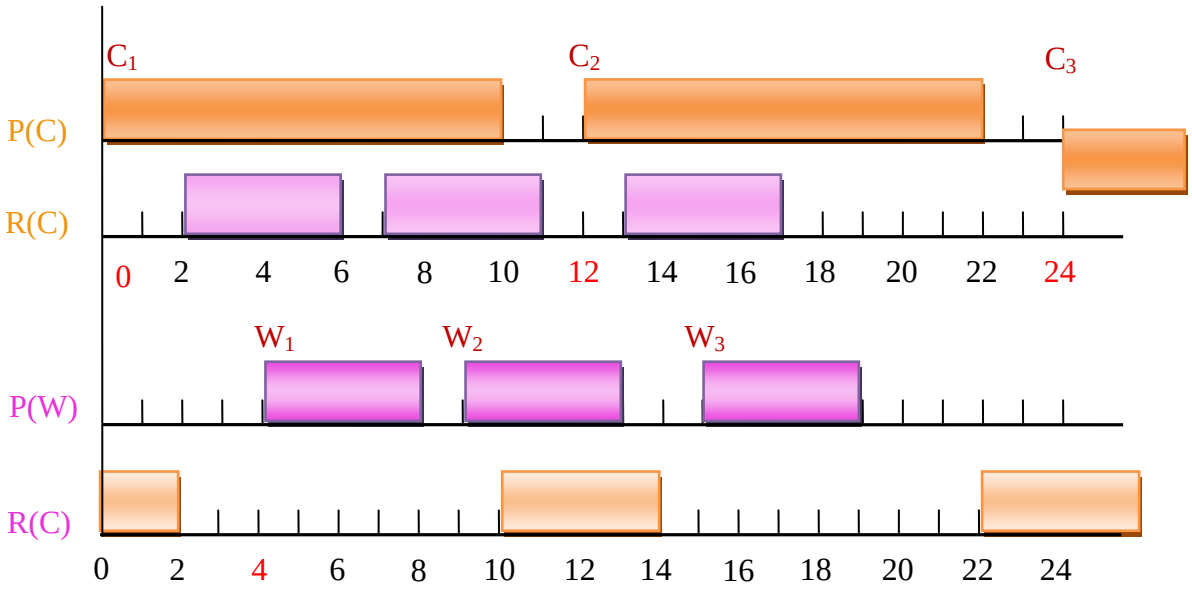


圖 6: 多個藥物的時程時間彈性

3.2.2.3 時間彈性之不確定性

前述的可服用彈性時間範圍存在不確定性。當某一劑量在它的彈性時間範圍內，在原本排定的時間點之前或之後服藥 (提早或延遲吃藥)，會影響它的前後劑量的可容許彈性時間範圍，同時也會影響與它有衝突的藥物的可服用時間範圍，也就是說這樣的彈性時間範圍會依照使用者實際服藥的時間動態改變。為了能夠根據劑量的服用時間彈性範圍預先做時程上的安排，我們分析彈性時間範圍的特性，避開動態改變的時間段，取其固定的時間段。

每一劑量的可服用時間範圍會被前後劑量的時程點所限制，同時也被前後衝突劑量的時程點所限制，這樣的時間範圍存在幾個特性：

1) 任何劑量的時程點一定包含在它的彈性時間範圍內。
2) 當某劑量在有效的彈性時間範圍內的任何一時間點服用，其前後劑量的時程永遠是合法的、有效的時程點。
3) 當某劑量在可提早的時間範圍服用時，不會縮小前後劑量的可提早時間範圍，會影響前後劑量的可延遲時間段 (可能會縮小原本的可延遲時間)。同理，當劑量延遲服

用時，會影響前後劑量的可提早時間。

4) 當劑量延遲服用時，會影響下一個衝突劑量的可提早時間範圍，當劑量提早服用時，會影響前一個衝突劑量的可延後時間範圍。

第一點是非常直觀的特性，OMAT 排出來的時程即為滿足所有服用限制的時程，而可服用的時間範圍，也是滿足所有限制的彈性時間，所以每一劑量的時程點一定包含在該劑量的可服用時間範圍內。

第二個特性的原因是因為任何劑量的彈性時間範圍都被前後的時程點所限制 (公式 (3), (4))，所以不管在彈性時間範圍內怎麼移動，前後劑量的時程點是合法的。我們可以推論，任一劑量在有效範圍內的提早或延遲服用，不會影響所有其他劑量的時程 (但是會影響前後劑量的彈性時間範圍)。

第三個特性，我們討論當某一劑量在其可服用時間範圍內移動，對其前後劑量的影響。由於藥物的劑量間，須滿足最大最小間隔限制，也就是說，同一藥物的任何連續劑量間，須保持在最小與最大距離之間。當某一劑量 A_i 提早服用，代表它遠離下一劑量 A_{i+1} 的時間，同時接近前一個劑量 A_{i-1} 的時間，所以下一劑量 A_{i+1} 可遠離它的距離縮小了 ($I^i(A_{i+1})$的可延遲時間)，可以接近它的距離增加了 ($I^i(A_{i+1})$的可提早時間)。A_i 的前一劑量 A_{i-1} 可接近它的距離縮小了 ($I^i(A_{i-1})$的可延遲時間)，可以遠離它的距離增加了 ($I^i(A_{i-1})$的可提早時間)，反之亦然。對交集後的可服用區間的影響，$I(A_{i+1})$的可延遲時間可能縮小或不變，可提早時間可能增加或不變；$I(A_{i-1})$的可延遲時間可能縮小或不變，可提早時間可能增加或不變。舉例來說，假設藥物 A 的服用頻次為 8 小時一次，時間間隔範圍為 [6, 10]，時程與每一劑量的可服用時間範圍如圖 7 (a)，當 A_3 從 16 提早到 15 服用，$I^3(A_2)$與 $I^3(A_4)$都減少一個單位的可延遲時間，交集後的結果 $I(A_2)$與 $I(A_3)$也減少一個單位的可延遲時間，可提早時間皆不變，如圖 7 (b)。當 A_3 從 16 延遲到 17 服用，則 $I^3(A_2)$與 $I^3(A_4)$都減少一個單位的可提早時間，交集後的結果 $I(A_2)$與 $I(A_4)$一樣也減少一個單位的可提早時間，但是可延後時間皆不變。當我們反向看這樣的關係，假設原本的時程為圖 7 (b)，A_3 從原定的時間點 15 延後到 16 服用，$I^3(A_2)$的可提早時間段縮小 (從[5, 8]

變成[6, 8])，但是交集後 I(A_2)的可提早時間段沒有改變 [6, 8]，I(A_2)的可延遲時間段增加 ([8, 9] -> [8, 10])，I(A_4)的可提早時間段也沒有改變，I(A_4)的可延後時間段也增長 1 單位。類似的現象也會發生在 A 的時程從圖 7 (c)變成圖 7 (a)的時候。

總而言之，當某一劑量延後服用時，它的前後劑量的可提早時間可能會縮小或不變，它的可延後時間可能會延長或不變。所以，當劑量延後服用時，其前後劑量原有的可延後時間段仍有效 (不會因此縮小原有的可延後時間)，同理，當劑量提前服用時，其前後劑量原有的可提早時間段仍有效 (不會因此縮小原有的可提早時間)。

第四個特性，由於衝突劑量之間須滿足衝突間隔限制，也就是衝突劑量之間至少須遠離衝突間隔時間單位，當劑量延遲服用時，代表它更接近下一個衝突劑量，所以下一個衝突劑量可以提前的時間就縮小了，但不影響下一劑量的可延後時間。同理，當某劑量提早服用時，則它的前一個衝突劑量的可延後時間就縮小了，不影響前一劑量的可提早時間。所以，衝突劑量之間亦滿足單向時間範圍不變的特性。所以我們可以利用單向的彈性時間範圍 (每個劑量的可延遲時間或每個劑量的可提早時間) 預先做時程上的處理。

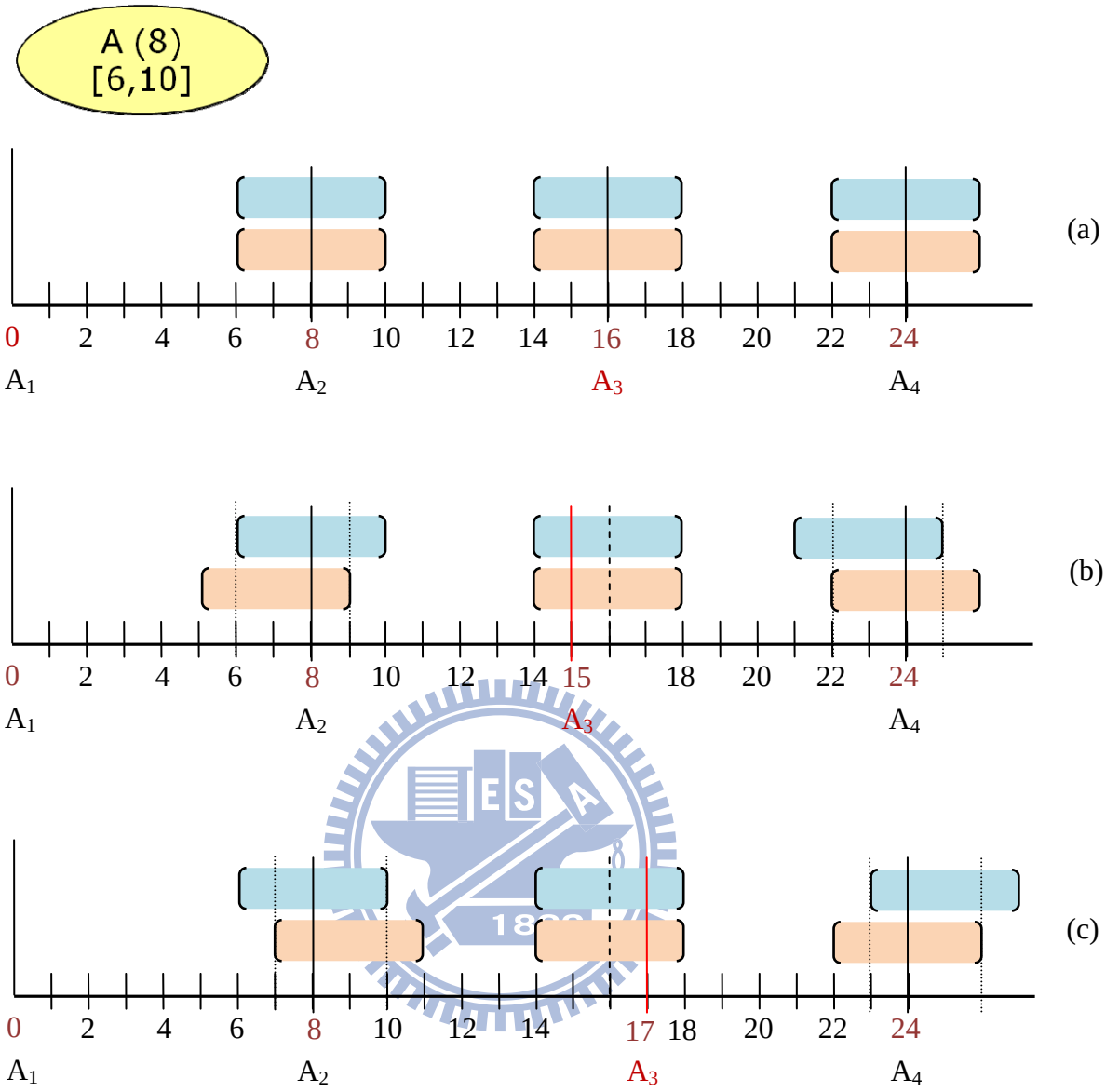


圖 7: 時間彈性的不確定性

第 4 章　藥物時程之服用延伸二 – 劑量分群

4.1　原理

我們取劑量的可延遲時間，做為每個劑量的可服用時間，可將時間段相交疊的劑量分在同一群 (group)，合併服用。下面章節中所用提到的可服用時間皆為前述的可延遲時間段。我們先將時程內的所有劑量依照其相互交集關係分成一到多個組別 (cluster)，然後計算每個類別內之劑量的所有可能的分群組合，再依照多維決策決定法，決定一個最佳的分群組合，每個類別所計算出來的最佳分群組合起來便是整個時程的最佳分群方式。節介紹我們將所有劑量分到一至多個類別的方式，4.2 節會介紹我們如何計算每個類別內之所有可能的分群組合，4.3 節會介紹我們如何利用多維決策決定法(MCDM)來決定最佳的分群組合。

4.2　所有可能之分群組合

4.2.1　劑量分組

我們將時程裡的所有劑量依照其可服用時間相互重疊的關係，分成一到多個組別。組別裡的任兩個劑量，不是有直接的交集，就是有間接的交集。所謂的直接交集，是指這兩個劑量的可服用時間有重疊；間接交集，是指這兩個劑量的可服用時間段沒有時間上的重疊，但是中間可能存在一到多個直接交集的關係，例如劑量 A_i 與劑量 C_j 沒有直接交集，但是 A_i 與 M_k 有直接交集，M_k 與 C_j 也有直接交集，則 A_i 與 C_j 為間接交集關係。

假設有 n 個劑量 $(d_1, d_2,\ldots,d_n)$，分成 m 個組別 $(S_1, S_2,\ldots,S_m)$

每一劑量皆滿足以下特性：

1) 每一個劑量皆屬於某一組別，且只屬於某一組別。

2) 每一劑量與該組別裡的其他任何劑量，一定存在直接或間接的交集關係，且至少與一個劑量有直接交集關係。

舉例來說，假設有三種藥 A,B,C，總共有 8 個劑量，服用時程及可服用時間如圖 8 所示，這八個劑量依彼此的交集關係分成下列三個組別：

$S_1 = \{d_1, d_2\}$

$S_2 = \{d_3, d_4, d_5, d_6\}$

$S_3 = \{d_7, d_8\}$

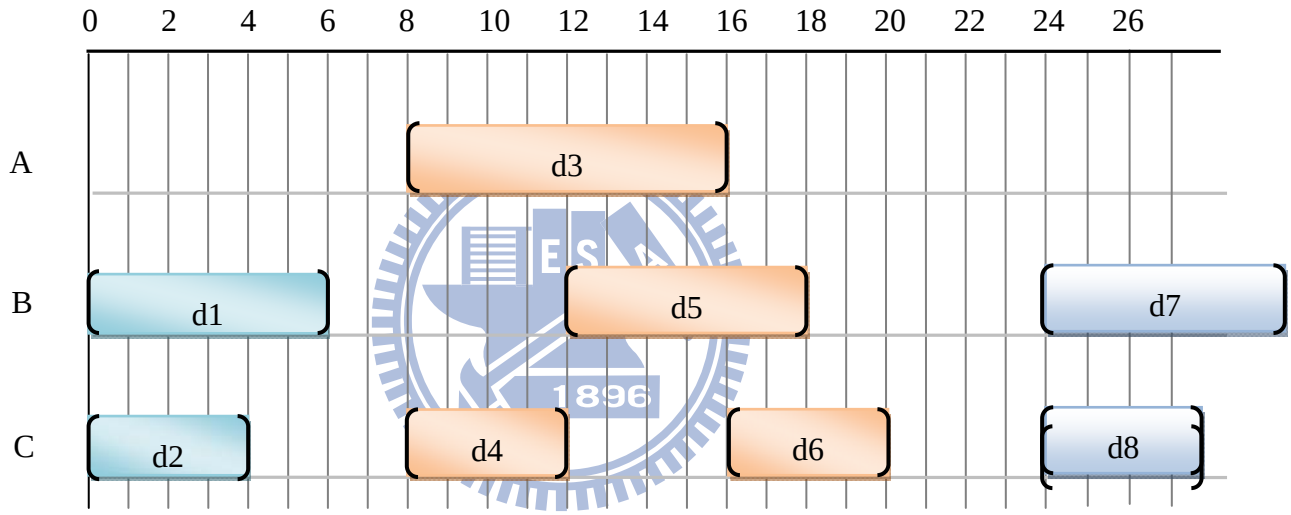


圖 8: 劑量分組圖例

4.2.2 基本服用時間單元和所有可能之分群組合

我們將劑量依照相互重疊的關係分到不同組別後，由於類別之間的劑量沒有任何時間交集關係，也就是彼此互相獨立，所以我們可以依照組別來做分群決定。在分群時，我們將服用時間段相重疊的劑量合併一起服用，但是一個劑量可能同時與多個劑量重疊於不同的時間段，這時候我們須考慮所有可能的分群方式 (group)。這一小節便討論如何找出每個類別內的所有可能的分群方式。

每一劑量(d_i)的可服用時間段會根據它與其它劑量的交集被切成一到多份連續的時間段

($d_{i1},\ldots,d_{ik}$)。每一小份我們稱為 d_i 的子時間段，以圖 8 的第二個組別 $S_2 = \{d_3, d_4, d_5, d_6\}$ 為例，這四個劑量分別可以被切成不同大小的子時間段，如圖 9 所示。d_3 被分成兩個子時間段，d_4 被分成一個子時間段，d_5 被分成兩個子時間段，d_6 被分成兩個子時間段。所有可能的分群方式即為所有可能的子時間段的組合，也就是子時間段的個數相乘，2x1x2x2 ＝ 8，如圖 10 所示，每一橫列代表一個可能的組合方式，框起來的部分代表時間段重疊，分在同一群 (group)，每一種組合方式即為該類別內的所有劑量的服用時間的組合，可能包含一到多個群，例如圖中的組合 1，包含了三個群，裡面共有四個劑量的服用時間，第一個群裡有 dose 3 和 dose4 (6 點到 10 點服用)，第二個群裡有 dose5 (10 點到 14 點服用)，第三個群裡有 dose6 (14 點到 16 點服用)。

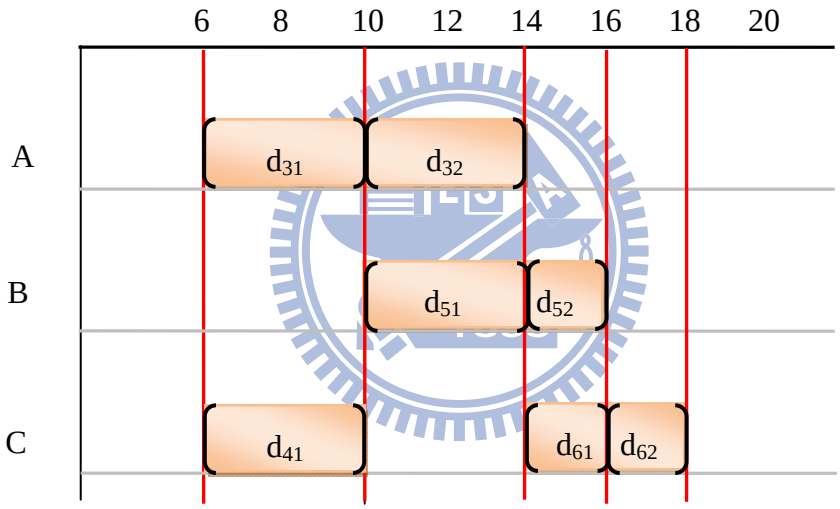


圖 9: 劑量基本服用時間單元

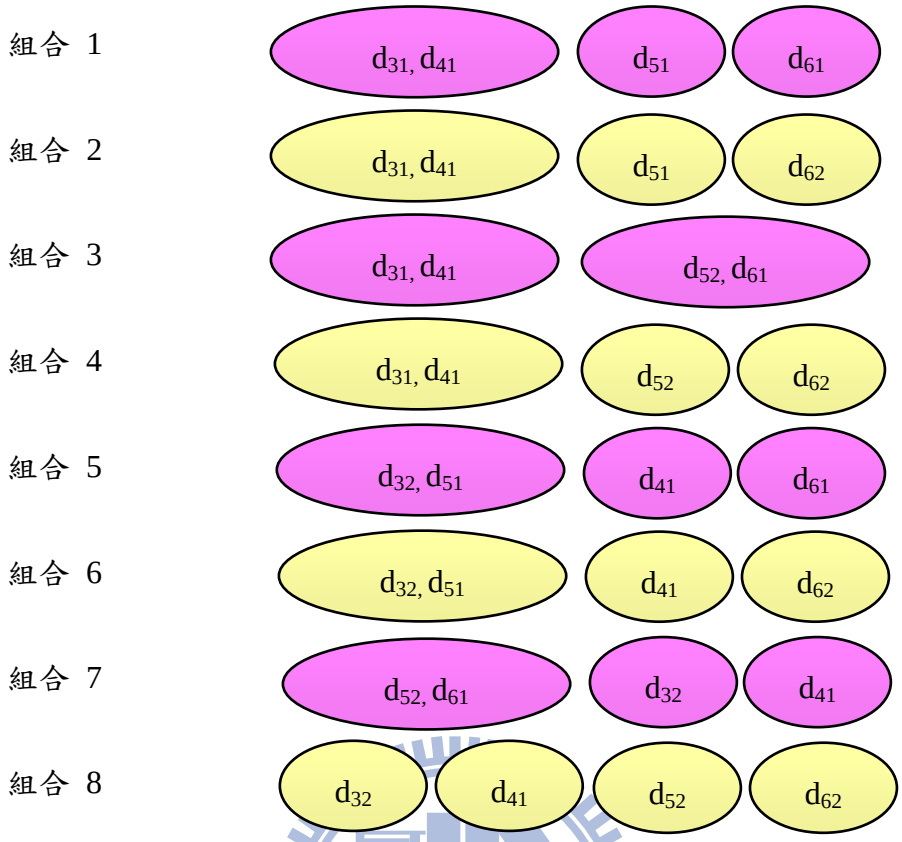


圖 10: 所有可能之劑量分群組合

4.3 找出最佳分群組合

我們利用多維決策決定法來選出每個組別內的最佳組合，每個組別的最佳組合結合起來便是整個時程的最佳組合。將我們的決定問題對應到多維決定方法，大致可分為幾個步驟，第一，先決定相對應的評估因素與選擇，第二，決定每個評估因素所占的比重，第三，量化每個選擇對應到每個評估因素的數值，第四，利用多維決策方法來評分，我們這裡選擇的是具代表性的線性方法，Revised AHP method 來評分，評分後得到最高分數的即是最佳選擇。下面將詳細介紹每個步驟的細節。

4.3.1 決定評估因素和選擇

我們的選擇即是每個組別內之劑量的所有可能組合。相對應的評估因素就是我們在選擇時所考量的因子。我們分群的目標是希望讓重要的藥有較大的服用機會，同時希望能減少服藥次數，也希望能讓服藥的時間有較大的彈性。我們針對這些目標訂定我們考量的因子。這些因子主要可分為兩類，一類是與分群 (group) 有關的因子，另一類是與劑量 (dose) 有關的因子。如表格 6 所示，第一行是與劑量有關的因子，包括這個劑量的重要性，劑量的可服用時間，劑量的服用時間被切成幾個子時間段。第二行是與劑量的子時間段有關的因子，包括子時間段的長度，該時間段與其他子時間段的順序關係；該子時間段的結尾離最後可以吃的時間 (deadline) 有多遠；錯過該子時間段的服藥時間後，後面還剩幾個子時間段 (還剩幾次機會可以服藥)。第三行是與分群有關的因子，包括每一群裡包含幾種藥，這個 group 的可服用時間長度，最後一個是與分群組合 (combination of groups) 有關的因子，每一個組合內總共有幾群。

劑量因子	劑量時間段因子	群組因子
可服用時間長度	時間段長度	群組服用時間長度
藥物重要性	剩餘時間段個數	劑量個數
時間段個數	距離期限的時間	群組個數

表格 6: 所有可能的評估因子

但是並非所有相關的因子都是我們在做多維決策決定時的評估因素，只有與目標相關的因子會被考慮，我們選擇的有效因子有三個，一個是與分群有關的，組合內的群組數目，另外兩個是與劑量的子時間段有關的因子，子時間段長度，以及剩餘的子時間段個數。舉例來說，圖 8 的第一類別 S1 = {d1, d2}的所有可能選擇有兩個 (d11, d21), (d12, d21)，d11 的時間是[0, 4]，d12 的時間是[4, 6]，d21 的時間是[0, 4]，評估因素則有五個，一個是組合內的群組數目，每一劑量分別有兩種與劑量相關之考量因子，因此決策矩陣的大小會隨選擇的多寡以及劑量個數而改變。S1 的決策矩陣如圖 11 所示。

分群組合	群組個數 (w1)	時間段長 (d1j) (w2)	剩餘次數 (d1j) (w3)	時間段長 (d2j) (w4)	剩餘次數 (d2j) (w5)
(d11, d21)					
(d12, d21)					

圖 11: 分群之決策矩陣

4.3.2 階層化決策因子的比重

4.3.2.1 階層化比較法

我們利用階層化 (Hierarchy) 的概念，來比較決策因子的相對重要性。每一層內的決策因子比較是利用成對比較法 (pairwise comparison)。它的概念是將兩兩評估因素相比，由決策者決定每一對評估因素的相對重要性，最後將它套用至數學的特徵方程式 [Saaty, 1980]，算出每個評估因素的比重。我們利用一個刻度尺(scale)來量化成對比較的重要性比率 (w_i / w_j)。這樣的刻度尺是一對一的對應關係 (one-to-one-mapping)，它將語言形態的重要性選擇對應到數值化的重要性比率，這樣的集合通常是離散的有限項集合。我們

使用 [Ma and Zheng, 1991]所提出來的刻度尺，如表格 7 所示。

每一層的所有評估因素的比較值可以用一個矩陣 A 表示，矩陣 A 的每一個元 a_{ij} 代表第 i 個 criteria 與第 j 個 criteria 的重要性比值 (w_i/w_j)。很明顯的 $a_{ij} = 1/a_{ji}$，所以 A 是一個倒值矩陣 (reciprocal matrix)。所以可以推導成下列的方程式[Saaty, 1980]：

$$\sum_{j=1}^{n} a_{ij} w_j = \sum_{j=1}^{n} w_i = n\, w_i \qquad (7)$$

for $i = 1, 2, \ldots, n$

或者表示成特徵方程式： $A\,W = n\,W$ (8)

n 為 criteria 的個數，對應到特徵值為 n 的特徵向量 W 即為每個評估因素的比重值。

相對比重	定義	說明
1/1	相等重要	兩比較因子的貢獻程度相等重要
9/7	稍微重要	稍微傾向於重視某一因子
9/5	頗重要	強烈傾向於重視某一因子
9/3	極重要	非常強烈傾向於重視某一因子
9/1	絕對重要	絕對傾向於重視某一因子
9/8, 9/6, 9/4, 9/2	相鄰尺度間的中間值	折衷選擇值
上列的倒數值	當比較角色顛倒時，會得到倒數數值	

表格 7: 相對比重的尺規 (Ma and Zheng, 1991, Satty,1980)

在階層化的比較中，我們將所有的決策因子分為三個階層，第一層，群組因子與劑量因子的比較，第二層，劑量與劑量之間的比較，第三層，劑量因子之間的比較。以 S_1 為例，第一層比較所有與分群有關的因子與所有與劑量有關的因子之間的相對重要性；第二層比較 dose 1 與 dose2 之間的相對重要性；第三層針對每個劑量比較劑量內的兩個因子之間的相對重要性，dose 1 的子時間段長與 dose 1 的剩餘次數的相對重要性，dose 2 的子時間段長與 dose 2 的剩餘次數的相對重要性。為了滿足關係的一致性，第三層的劑量因子比重和會滿足第二層的劑量間相對比重關係，第二層的劑量比重和會滿足第一層的比重相對關係。階層化比重值計算的程序可分為幾個步驟：

步驟 1)

第一層的比較，比較群組因素 (Group Criteria) 與劑量因素 (Dose Criteria)的相對重要性，它的比重比較矩陣 $A_{G/D}$ 如下所示，$a_{ij} = w_i / w_j$。

$$A_{G/D} = \begin{array}{c} \\ G \\ D \end{array} \begin{array}{c} \begin{array}{cc} G & D \end{array} \\ \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \end{array}$$

步驟 2)

利用特徵方程式 $\mathbf{A}W = n\,W$ 計算出當 n 為 2 時所對應的特徵向量。w_1 是群組因素的比重與 w_2 劑量因素的比重。

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = 2 \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

$$\mathbf{A_{G/D}} \quad W = n\,W$$

步驟 3)

正規化步驟 2 求出來的特徵向量 W，將向量裡每一元的值除以向量裡所有元的值的和。

$$\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \xrightarrow{\text{normalize}} \begin{pmatrix} \dfrac{w_1}{w_1 + w_2} \\ \dfrac{w_2}{w_1 + w_2} \end{pmatrix} = \begin{pmatrix} w_G \\ w_D \end{pmatrix}$$

$$W \qquad\qquad W_N$$

步驟 4)

第二層的比較，比較兩兩劑量之間的相對重要性，它的比較矩陣 A 如下圖所示。

$$A_{D/D} = \begin{array}{c} \\ d_1 \\ d_2 \\ . \\ . \\ d_n \end{array} \begin{array}{c} \begin{array}{cccccc} d_1 & d_2 & . & . & . & d_n \end{array} \\ \begin{pmatrix} a_{11} & a_{12} & . & . & . & a_{1n} \\ a_{21} & a_{22} & & & & . \\ . & & & & & \\ . & & & & & \\ a_{n1} & a_{n2} & . & . & . & a_{nn} \end{pmatrix} \end{array}$$

步驟 5)

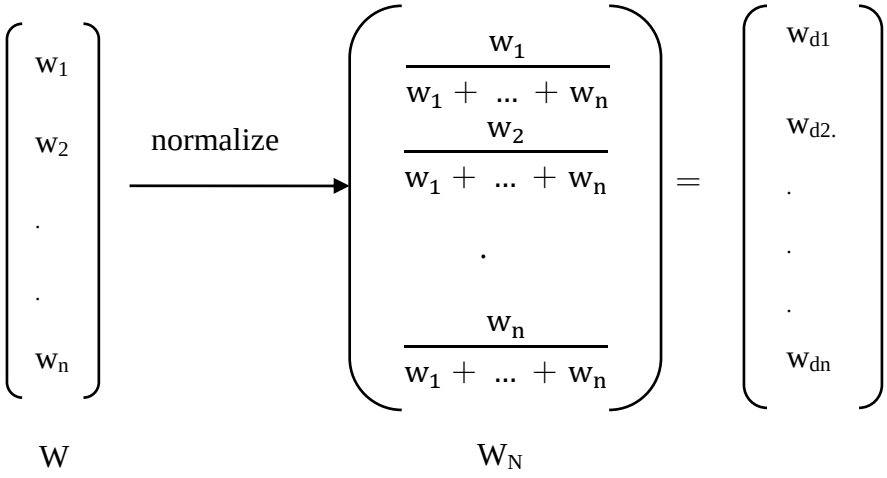
利用特徵方程式 **A**W = n W 計算特徵值為 n (dose 的個數)相對應的特徵向量。

$$\mathbf{A_{D/D}} \begin{pmatrix} w_1 \\ w_2 \\ . \\ . \\ w_n \end{pmatrix} = n \begin{pmatrix} w_1 \\ w_2 \\ . \\ . \\ w_n \end{pmatrix}$$

$$\mathbf{A_{D/D}} \quad W \quad = \quad n \quad W$$

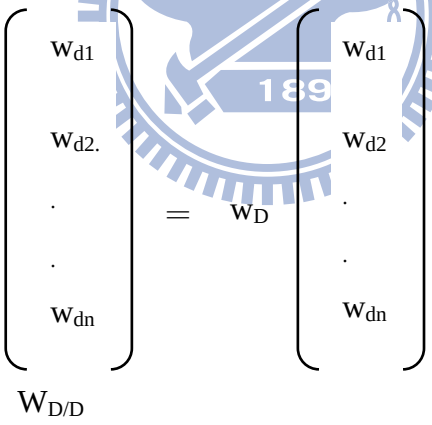
步驟 6)

正規化步驟 5 求出來的特徵向量 W，將向量裡每一元的值除以向量裡所有元的值的和。



步驟 7)

將正規化後的特徵向量乘上步驟 3 所算出來的劑量比重值 w_D，乘完後的值才是劑量真正的比重值。



步驟 8)

第三層的比較，比較劑量的子時間段長度與剩餘次數的相對重要性，每一個劑量都有一個這樣的矩陣，n 個劑量就有 n 個矩陣。

$$A_{Df/Df} = \begin{array}{c} \\ d_i\text{-duration} \\ d_i\text{-time} \end{array} \begin{array}{c} \begin{array}{cc} d_i\text{-duration} & d_i\text{-time} \end{array} \\ \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \end{array}$$

步驟 9)

利用特徵方程式 **A**W = n W 計算特徵值為 2 時相對應的特徵向量。

$$\mathbf{A_{Df/Df}} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = 2 \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

$$\mathbf{A_{Df/Df}} \quad W = 2W$$

步驟 10)

正規化步驟 9 所求出的特徵向量 W

$$\underset{W}{\begin{pmatrix} w_1 \\ w_2 \end{pmatrix}} \xrightarrow{\text{normalize}} \underset{W_N}{\begin{pmatrix} \frac{w_1}{w_1 + w_2} \\ \frac{w_2}{w_1 + w_2} \end{pmatrix}} = \begin{pmatrix} w_{\text{di-duration}} \\ w_{\text{di-time}} \end{pmatrix}$$

步驟 11)

將步驟 10 正規化後的向量乘上該劑量的比重 w_{di} (步驟 7 的結果)。

$$\begin{pmatrix} w_{di\text{-}duration} \\ w_{di\text{-}time} \end{pmatrix} = w_{di} \begin{pmatrix} w_{di\text{-}duration} \\ w_{di\text{-}time} \end{pmatrix}$$

步驟 12)

重複(步驟 9, 10, 11) n 回合，n 為 dose 的個數。最後可得到每個評估因素的比重值，如下所示。

$$\begin{pmatrix} w_G \\ w_{d1\text{-}duration} \\ w_{d1\text{-}time} \\ . \\ w_{dn\text{-}duration} \\ w_{dn\text{-}time} \end{pmatrix}$$

運用上述的機制，可以階層化的比較相對重要關係，推導出所有的決策考量因子的比重 (weight of criteria)。有了這些資訊，便可建立我們的決策矩陣，然後利用 Revised AHP method 來為每個選擇評分，進而得到較佳的選擇。

4.3.2.2 藥物相對重要性分類

我們設計了一套機制來降低決策者在比較決策因子之相對重要性的複雜度。我們定義了三類的藥物重要性，讓醫藥人員能夠很簡單將病人的藥物依照不同的緊急程度 (重要性) 分類。然後根據藥物的重要性，轉換成與劑量相關的評估因子相對比重。我們設計的分類方式如下：

Class 1) 重要藥物：意指該藥物對病人的病情或健康控制有極大的影響，病人忘記服 藥可能有嚴重的危險，像是控制高血壓、心臟病等藥物。

Class 2) 一般藥物：意指該藥物偶爾忘記服用不會有嚴重的危險。

Class 3) 非必要性藥物：意指該藥物是需要時服用的藥 (take when need)，也就是 PRN (pro re nata) 用藥，像是退燒藥或頭痛藥等，當病人覺得不舒服或出現某些症狀時才需服藥。

不同類別的藥物之間的相對重要性，如下表所示：

藥物重要性比較	相對重要性	相對比重
重要藥物 vs. 一般藥物	頗重要	9/5
一般藥物 vs. 非必要性藥物	頗重要	9/5
重要藥物 vs. 非必要性藥物	極重要	9/3
同一類藥物	相等重要	1/1

表格 8: 分類藥物之相對重要性尺規

我們可以利用這些分類，自動轉換成上一小節介紹的階層化比較的第二層，劑量之間的重要性比較。關於第三層劑量內評估因素比較，我們針對不同重要性的藥有不同的考量。重要藥物通常不太能容許病人忘記服藥，所以對於這類的藥物，我們希望能有較大的服用機會，所以在分群時，我們會希望安排在剩餘子時間段 (remaining slots) 較多的時間段內，時間的彈性為次要的考量。對於一般藥物，基本上有容許病人少次錯過藥的彈性，所以我們會以時間彈性長度為主要考量，時間長度大，病人在服用時間上的彈性較大，系統重排的機率較低，但是當病人錯過此次服藥後，後面可以再吃的機率較低。對於需

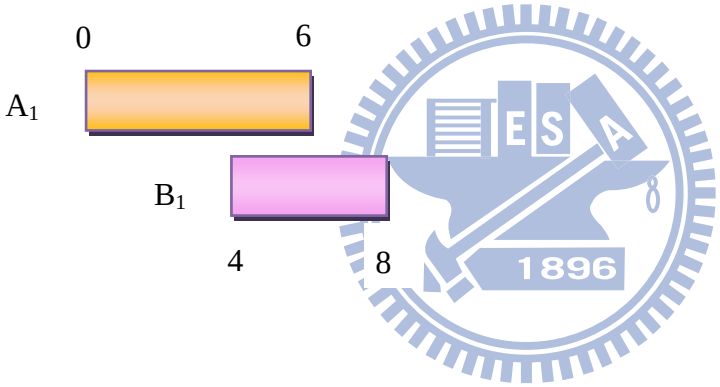
要時服用的藥物，我們將這兩個因子視為一樣重要。這些對應關係如下表所示：

藥物分類	時間段長 vs. 剩餘次數	相對比重
重藥劑量	稍微重要	7/9
一般劑量	稍微重要	9/7
非必要性劑量	相等重要	1/1

表格 9: 分類劑量因子之相對重要性尺規

下面我們介紹一個簡單的例子，幫助你了解整個轉換還有運算的過程，同時我們比較不同重要性的運算結果。

假設有 A, B 兩種藥物，A 藥為第一類重要藥物，B 藥為第二類一般藥物，排程後 A_1 , B_1 ，可服用時間分別是[0, 6]和[4, 8]，如下圖。



按照階層化的比較方法，我們可以推導出決策矩陣中每個評估因素的比重。

步驟 1) 群組因素和劑量因素比較，我們假設群組因素與劑量因素

一樣重要，所以相對重要性矩陣為：

$$A_{G/D} = \begin{array}{c} \\ G \\ D \end{array} \begin{array}{c} \begin{array}{cc} G & D \end{array} \\ \begin{pmatrix} 1/1 & 1/1 \\ 1/1 & 1/1 \end{pmatrix} \end{array}$$

步驟 2) 利用特徵方程式解 W

$$\begin{pmatrix} 1/1 & 1/1 \\ 1/1 & 1/1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = 2 \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.7071 \\ 0.7071 \end{pmatrix}$$

$$\mathbf{A_{G/D}} \quad W \quad = \quad n \quad W \quad W$$

步驟 3) 正規化 W

$$\begin{pmatrix} 0.7071 \\ 0.7071 \end{pmatrix} \xrightarrow{\text{normalize}} \begin{pmatrix} \frac{0.7071}{0.7071+0.7071} \\ \frac{0.7071}{0.7071+0.7071} \end{pmatrix} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

$$W \qquad W_N$$

步驟 4) 比較劑量之間的重要性，從藥物的分類自種轉換。

$$A_{D/D}= \begin{array}{c} \\ A_1 \\ B_1 \end{array} \begin{array}{c} \begin{array}{cc} A_1 & B_1 \end{array} \\ \begin{pmatrix} 1/1 & 9/5 \\ 5/9 & 1/1 \end{pmatrix} \end{array}$$

步驟 5, 6) 利用特徵方程式計算 W 然後正規化。

$$\begin{pmatrix} 1/1 & 9/5 \\ 5/9 & 1/1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = 2 \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.8742 \\ 0.4856 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.6429 \\ 0.3571 \end{pmatrix}$$

$$\mathbf{A_{D/D}} \quad W \quad = \quad n \quad W \quad W \quad W_N$$

步驟 7) 將步驟 6 算出來的 W，乘上劑量的比重值 (步驟三)

$$\begin{pmatrix} w_{d1} \\ w_{d2} \end{pmatrix} = \begin{pmatrix} 0.3215 \\ 0.1786 \end{pmatrix} = 0.5 \begin{pmatrix} 0.6429 \\ 0.3571 \end{pmatrix}$$

$W_{level\text{-}2}$ $\quad$ w_D $\quad$ W_N

步驟 8-11) 比較劑量內因子的相對重要性，這裡有兩個劑量，所以重複兩個回合。

A_1 的評估因子比重:

$$A_{level\text{-}3\text{-}A1} = \begin{matrix} \\ d_i\text{-duration} \\ d_i\text{-time} \end{matrix} \begin{matrix} \begin{matrix} d_i\text{-duration} & d_i\text{-time} \end{matrix} \\ \begin{pmatrix} 1/1 & 7/9 \\ 9/7 & 1/1 \end{pmatrix} \end{matrix}$$

$$\begin{pmatrix} 1/1 & 7/9 \\ 9/7 & 1/1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = 2 \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.6139 \\ 0.7894 \end{pmatrix}$$

$A_{level\text{-}3\text{-}A1}$ $\quad$ W $\quad = \quad$ n $\quad W$ $\quad\quad$ W

$$\begin{pmatrix} 0.6139 \\ 0.7894 \end{pmatrix} \xrightarrow{\text{normalize}} \begin{pmatrix} 0.4375 \\ 0.5625 \end{pmatrix} \xrightarrow{\times\, w_{d1}\ (0.3215)} \begin{pmatrix} 0.1407 \\ 0.1808 \end{pmatrix}$$

W $\quad\quad\quad\quad$ W_N

B_1 的評估因子比重:

$$A_{\text{level-3-B1}} = \begin{array}{c} \\ d_i\text{-duration} \\ d_i\text{-time} \end{array} \begin{array}{c} \begin{array}{cc} d_i\text{-duration} & d_i\text{-time} \end{array} \\ \begin{pmatrix} 1/1 & 9/7 \\ 7/9 & 1/1 \end{pmatrix} \end{array}$$

$$\begin{pmatrix} 1/1 & 9/7 \\ 7/9 & 1/1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = 2 \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.7894 \\ 0.6139 \end{pmatrix}$$

$$\mathbf{A_{level\text{-}3\text{-}B1}} \quad W = n \quad W \qquad W$$

$$\begin{pmatrix} 0.7894 \\ 0.6139 \end{pmatrix} \xrightarrow{\text{normalize}} \begin{pmatrix} 0.5625 \\ 0.4375 \end{pmatrix} \xrightarrow{\times\ w_{d2}} \begin{pmatrix} 0.1005 \\ 0.0781 \end{pmatrix}$$

$$W \qquad W_N \qquad 0.1786$$

步驟 12) 每個評估因子的比重值 (小數後兩位，四捨五入)

$$\begin{pmatrix} w_G \\ w_{d1\text{-}duration} \\ w_{d1\text{-}time} \\ w_{d2\text{-}duration} \\ w_{d2\text{-}time} \end{pmatrix} = \begin{pmatrix} 0.50 \\ 0.14 \\ 0.18 \\ 0.10 \\ 0.08 \end{pmatrix}$$

4.3.3 效能之數值化

由於我們的評估因素裡沒有抽象的因子，因此可以很具體的用數值來表示，唯一要注意的是量化後的值，是否符合多維矩陣的意義。我們希望在每一行的元素裡，值越大的表示相對應的選擇在這個評估因素底下表現較好，也就是我們希望它的數值與表現的好壞成正比 (值越大的越好)。以 S1 為例，轉換出來的決策矩陣如圖 12。子時間段的長度是越長越好，越長代表該劑量在子時間段內服用的時間彈性越大，子時間段的剩餘次數是越多越好，代表劑量錯過這個子時間段後，還可以再服用的機會越高，群數是越少越好，因為每一群代表一次的服藥，數量越少代表使用者須吃藥的次數越少，所以我們必須將群數的值取倒數 $\frac{1}{(\text{No.of Groups})}$ 。

另外，如果我們選擇的決策決定法為非線性方法，例如第二章所介紹的 WPM method，則矩陣裡的每個元的值不能為零，否則它將導致該選擇的分數永遠為零，而看不到其他評估因子對這個選擇的影響，這個情況的處理方法可以將該行的評估因素值加上一個特定的常數值，來避免數值為零的情形。調整後的值如圖 13 所示，我們將有 0 值的行的每一個元的值加常數值 1，然後將每一行的值除上該行的最大值。

Alts.	群組個數 (w_1)	時間段長 (d_{1j}) (w_2)	剩餘次數 (d_{1j}) (w_3)	時間段長 (d_{2j}) (w_4)	剩餘次數 (d_{2j}) (w_5)
(d_{11}, d_{21})	1	4	1	4	0
(d_{12}, d_{21})	2	2	0	4	0

圖 12: 決策矩陣的效能數值

分群組合	群組個數 (w_1)	時間段長 (d_{1j}) (w_2)	剩餘次數 (d_{1j}) (w_3)	時間段長 (d_{2j}) (w_4)	剩餘次數 (d_{2j}) (w_5)
(d_{11}, d_{21})	1/2	4/4	2/2	4/4	1/1
(d_{12}, d_{21})	2/2	2/4	1/2	4/4	1/1

圖 13: 調整後的矩陣效能數值

4.3.4 選擇的評分

我們利用 2.3.3 節介紹修正後之層級分析法的方法，來為每種選擇評分，分數的高低代表選擇的好壞。評分方式如下：

$$A_{score} = \max_i \sum_{j=1}^{n} a_{ij}\, w_j$$

這裡的 A 指的是決策矩陣，a_{ij} 是決策矩陣裡的每個相對應元的值。在 Revised AHP method 裡，它將原本矩陣裡每一行的數值，除上該行的最大值，消除不同維度下的量化單位的影響。假設我們最後得到的決策矩陣如表格 10。

我們可以算出 Alternative 1 和 Alternative 2 的分數：

$A_{1,\,score}$

$= (1/2)\times0.56 + (2/2)\times0.12 + (2/2)\times0.16 + (4/4)\times0.09 + (1/1)\times0.07 = 0.72$

$A_{2,\,score}$

$= (2/2)\times0.56 + (2/4)\times0.12 + (1/2)\times0.16 + (4/4)\times0.09 + (1/1)\times0.07 = 0.86$

分群組合	群組個數 (0.56)	時間段長 (d_{1j}) (0.12)	剩餘次數 (d_{1j}) (0.16)	時間段長 (d_{2j}) (0.09)	剩餘次數 (d_{2j}) (0.07)
(d_{11}, d_{21})	1/2	4/4	2/2	4/4	1/1
(d_{12}, d_{21})	2/2	2/4	1/2	4/4	1/1

表格 10: 決策矩陣範例

4.4 實際案例

我們以一個實際的藥物案例，來測試分群後的結果，我們選擇兩種互相有衝突劑量的藥物，以及三種不同服藥頻次的藥物來做分群實驗。這些藥物的相關參數如表格 11。

藥物名稱	Rosuvastatin	Magnesium Oxide	Spironolactone	Amlodipine	Propranolol
頻次	一天一次	一天三次	一天一次	一天兩次	一天三次
服藥週期	24	8	24	12	8
最小時間間隔	12	4	12	6	4
最大時間間隔	36	16	36	18	16
衝突藥物	Magnesium Oxide (2, 2)	Rosuvastatin (2, 2)	無	無	無
服用劑量限制	無	無	無	無	有
重要性分類	一般藥物	一般藥物	一般藥物	重要藥物	一般藥物

表格 11: 實際藥物案例的相關參數

下面是這五種藥的排程結果，以及每個劑量的可服用時間段，我們可以將所有劑量依照時間重疊關係分成兩個群組。

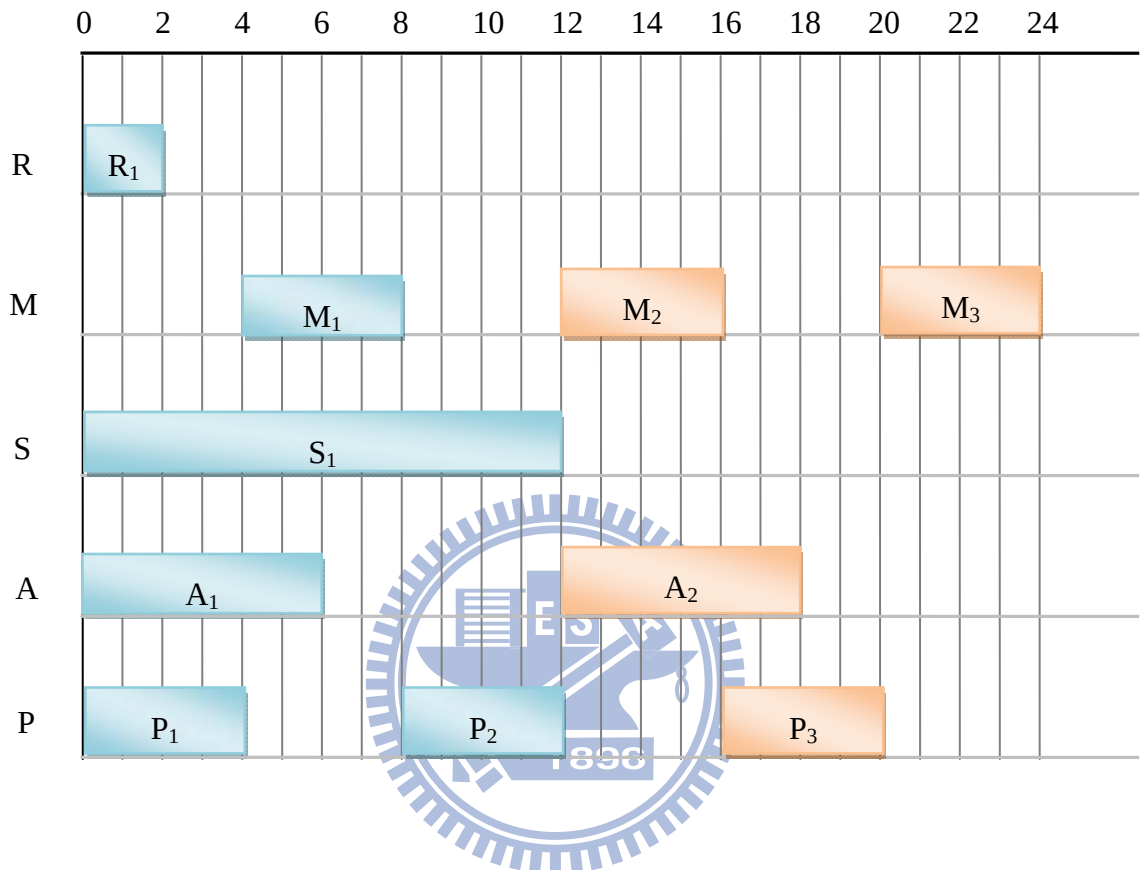


圖 14: 實際藥物的彈性時程範例

利用我們設計的分群方法，可以找到最佳的分群組合，如下圖:

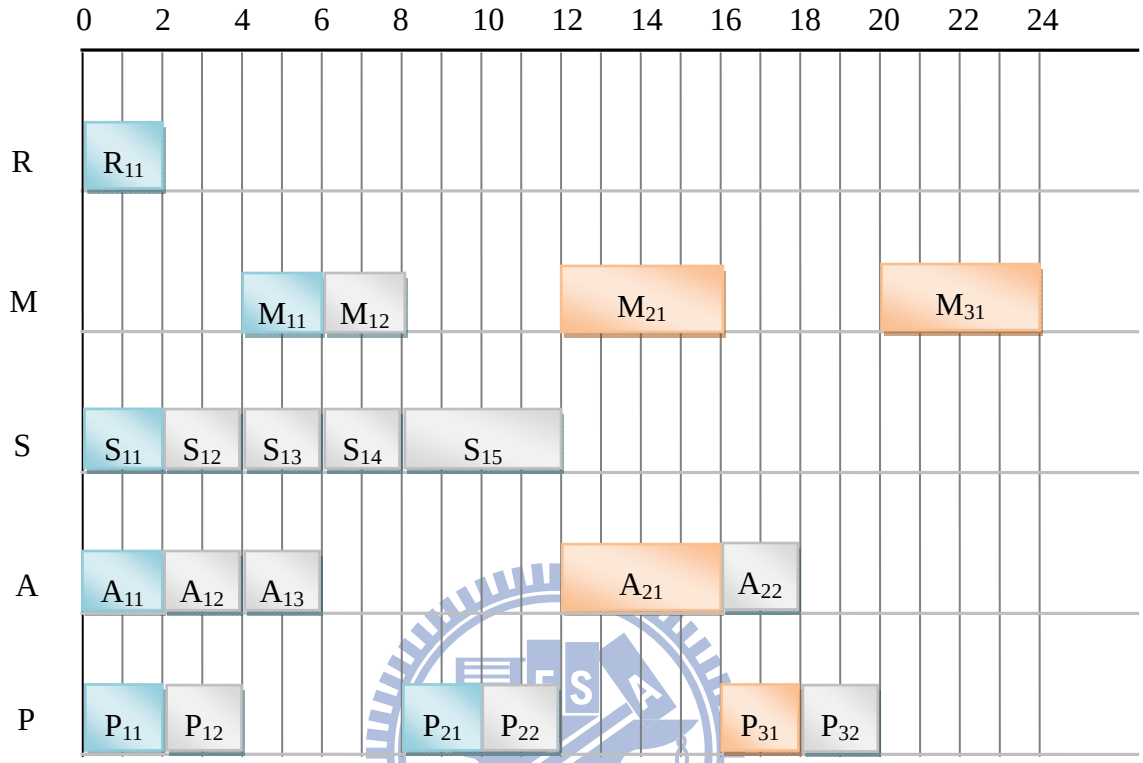


圖 15: 實際藥物的最佳分群組合

當使用者限制單次服藥數量時，我們仍可從現有的所有分群選擇中，找到符合限制的最佳選擇，下面是當劑量數量小於 4 時的最佳選擇。

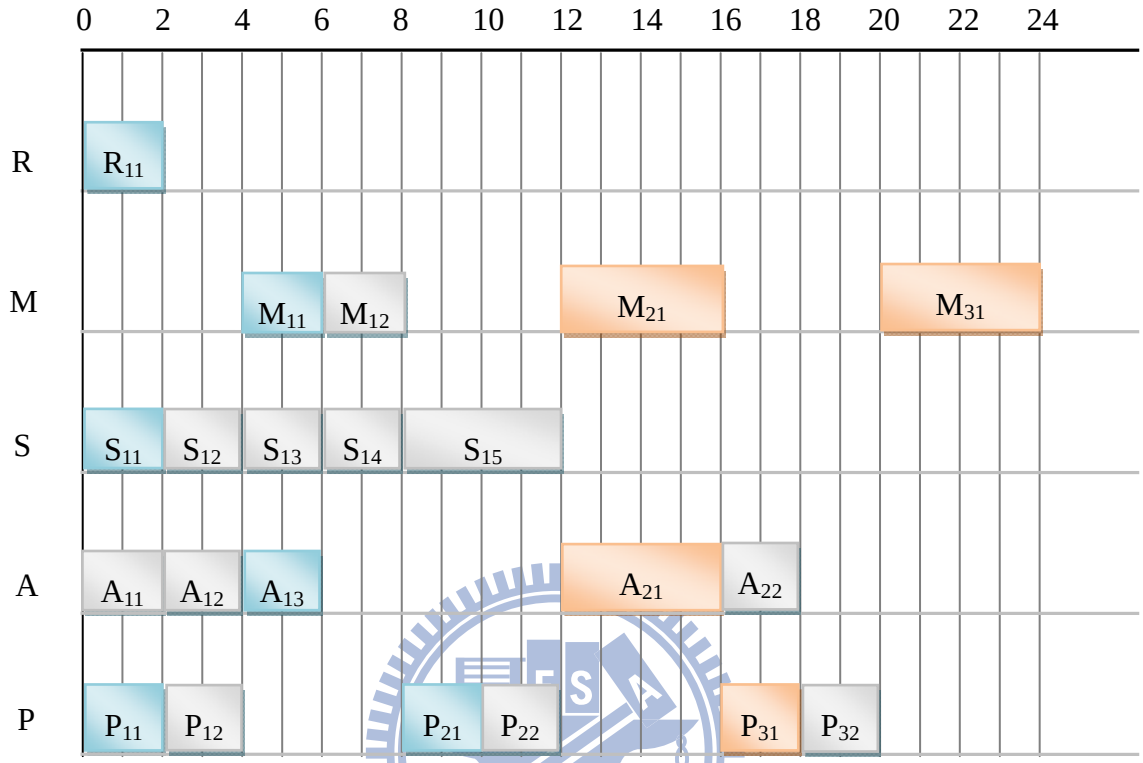


圖 16: 實際藥物之滿足劑量數目限制的最佳組合

當使用者錯過第一群的劑量時，我們仍可找到下一個最佳的分群組合，方法是從現有的分群組合中，忽略有 S_{11} 和 P_{11} 的組合，忽略每個組合內的 R_{11}，即可找到修改後的最佳組合，如下圖。

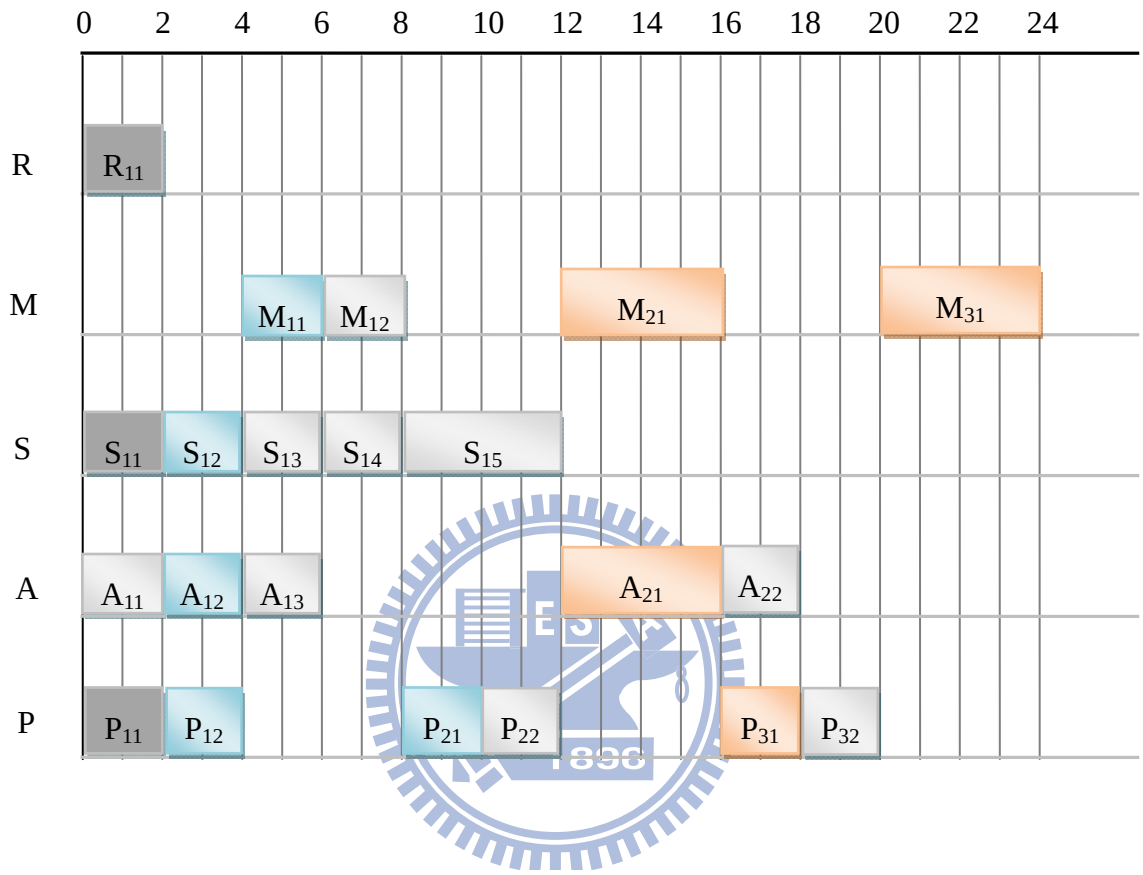


圖 17: 錯過群組藥物之修正組合

第 5 章　實作議題

5.1　實作原理

Wedjat 在.Net Framework 上開發，執行的作業平台環境為 Windows Mobile 6.0，並與手機內建的 Calendar 應用程式結合。

從圖 18 可以清楚看到 Wedjat 的資料流，Wedjat 輸入的資料是使用者的處方籤和使用者的喜好設定，用藥指示則是預先儲存在手機上的資料庫裡。Wedjat 的排程結果將透過 Calendar API 寫入手機的 Calendar 裡，當使用者按下服藥的確認鍵後，Wedjat 將輸出使用者的用藥紀錄，並定期上傳至遠端個人健康紀錄系統。

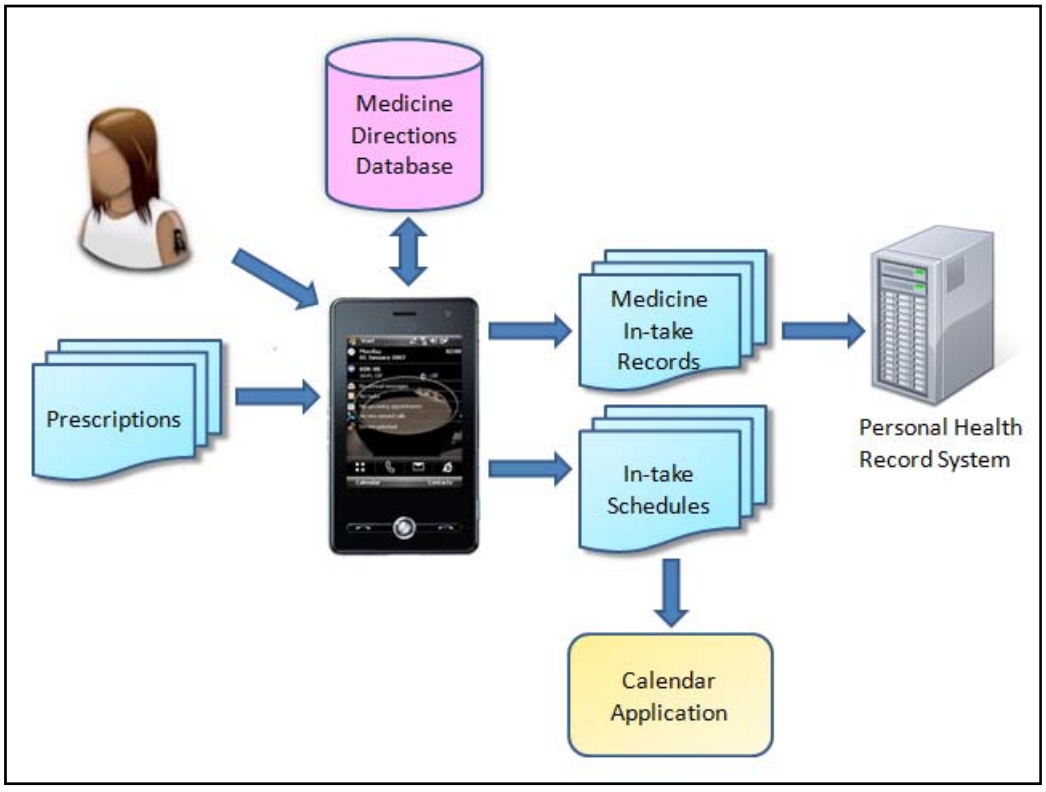


圖 18: Wedjat 輸入輸出資料流

5.2 操作方案

Wedjat 將時程裡的每一群組的劑量，視為一鬧鈴觸發事件，輸出至行事曆。預設在服藥時間的前 15 分鐘提醒使用者。當使用者確認服藥，Wedjat 會儲存使用者的服藥時間與內容。當使用者延遲吃藥時，鬧鈴會持續提醒，直到服藥的時程無效，或使用者關掉鬧鈴為止。當使用者錯過群組的服藥時間，排程器會動態的修改時程，並且更新行事曆的內容。如果使用者持續一段時間都沒有服藥，導致低於有效劑量時，Wedjat 會發出警示訊息，建議使用者聯絡醫生或諮詢藥師。

5.3 系統設計

Wedjat 的功能模組如圖 19 所示。Wedjat 的核心是藥物排程器 (Medication Scheduler)，負責安排使用者的服藥時程，動態更新修改時程。Wedjat 前端的功能模組負責處理輸入的資料，Wedjat 從遠端系統接收使用者的處方籤 (xml 檔案格式)，然後經由 XML Parser 解析資料，再經由 SQL Server Mobile 將資料輸入至手機的資料庫，與資料庫內存的用藥指示整合為 MSS，所以 SQL 裡存有排程所需的資訊 MSS 以及藥物相關的資訊、注意事項等等。排程器排程時，會向 SQL Server Mobile 查詢藥物的 MSS 資訊。Wedjat 後端的功能模組負責處理輸出的資料，排程器會將排程的結果，經由 Calendar API 寫入行事曆，並監控使用者的服藥行為，做動態的反應。如果使用者按下服藥確認鍵，排程器會輸出服藥紀錄。如果使用者錯過服藥，排程器會動態的修改時程，更新 Calendar 的內容，或者對使用者發出相對應的警示訊息。

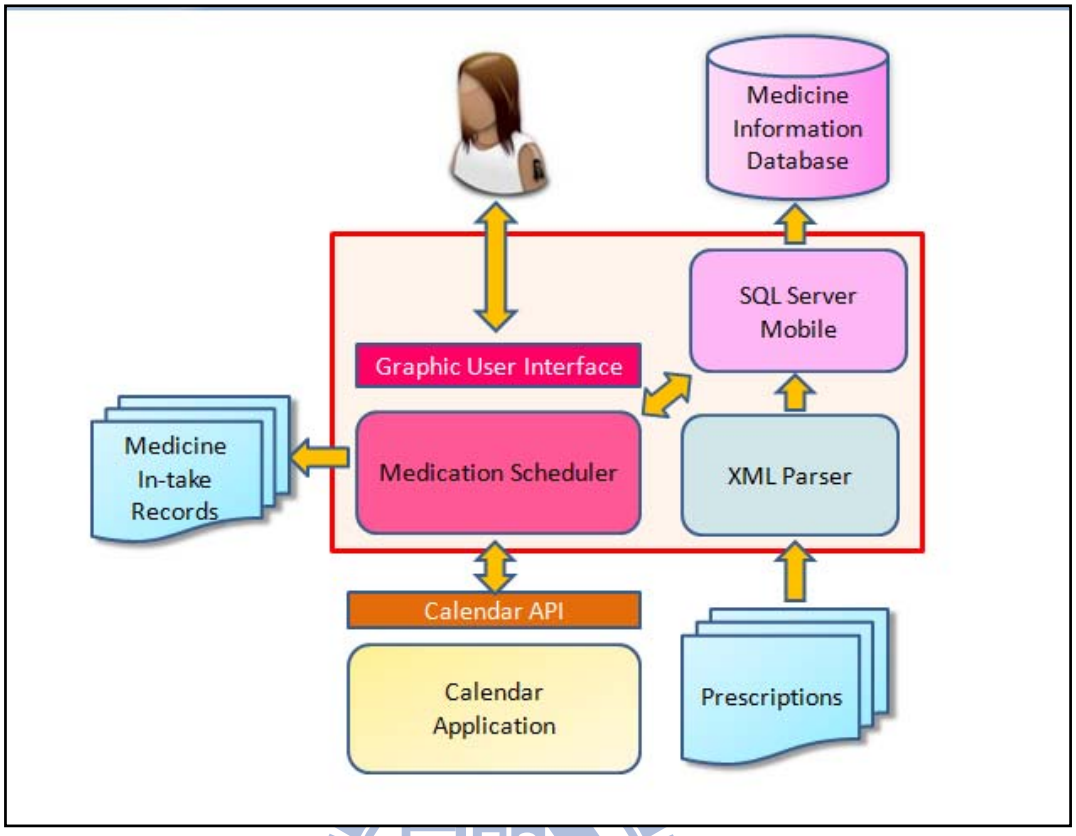


圖 19: Wedjat 功能模組

Wedjat 的類別物件 (class diagrams) 分為兩大類: Medication Scheduling Specification，Medication Scheduler。圖 20 的內容是我們實作 MSS 的資料結構，用來處理藥物相關的資訊與服用規則。藥物排程器的模組細節如圖 21 所示，這個部分是整個 Wedjat 的核心，負責運算所有藥物的時程。它為每種藥物產生一個 `JobModel` 和一個 `ResourceModel` 物件，這部分是實作 Resource Model 的概念，每一個 `JobModel` 物件裡會有每一種藥目前要排程的工作 (job) 的資訊，也就是下一個需要被排程的劑量，`deadline` 的初始值為 `nsmin`，之後的值為上一個工作的起始時間加上 `asmax`。 `ResourceModel` 物件則是用來確保藥物的劑量間的時間間隔與衝突劑量間的時間間隔，它擁有兩個整數陣列 *resource* 和 *processor*，陣列裡的每一個元的初始值為 0，當某一工作被排程時，會占用 *processor* k 個時間單位，k 為該工作執行的時間，所以會從該工作的起始時間往後占用 k 的時間單位，這段時間的 *processor* 會被設為 1。當被排程的藥物有衝突藥物時，它會占用衝突藥物的 *resource*，從該工作的起始時間往前占用 minFrInterferer，往後占用

minToInterferer，也就是將該時間段的 *resource* 設為 1。只有在 *processor* 和 *resource* 皆為 0 時，工作才能被開始執行。MIS 是藥物排程器輸出的服藥時程，是一個 List 的資料型態，每一個紀錄藥物的 ID，需服用的時間與劑量。

Medicine Priority 實作了第二章介紹的 5 種藥物優先順序的決定方式，EDF 只適用於 ODAT 演算法。

```
Class MSS{
    int med_id; DosageParameters dp;
    List<InteractionParameters> ip;
}
Class DosageParameters{
    int T;
    int nsmin, nsmax, asmin, asmax;
    inr dmin, dmax, Dmin, Dmax;
    int B, R, L, P;
}
Class InteractionParameters{
    int med_id; int mintoInterferer;
    int minFrInterferer;
}
```

圖 20: 藥物排程規格書相關類別

```
Class MedicationScheduler{
    int feasibleSchedule;
    public bool OMAT(List<MSS> mss,
        PrioritySchemes ps, ref List<MIS> mis);
    public bool ODAT(List<MSS> mss,
        PrioritySchemes ps, ref List<MIS> mis);
    private void SetDoseSize(List<MSS> mss, ref
            List<MIS>    mis);
    private void SetProcessor(int latestStartTime,
        MSS mss,    ref ResourceModel rm);
    private void SetResource(int med_id,
        List<MIS> mis,  List<MSS> mss,
        ref ResourceModel rm);
}
Class ResourceModel{
    int[] resource = new int[T];
    int[] processor = new int[T];
    bool feasible = TRUE; List<int> schedule = NULL;
    int priority = 0;
}
Class JobModel{
    int release_time = 0;
    int execution_time = ns_min;
    int deadline = release_time + execution_time;
}
Class MIS{
    int med_id; int time;   int dose_size;
}
Enum PrioritySchemes{RM, MVF, MIF, SSDF, EDF};
```

圖 21: 藥物排程相關類別

5.4 使用者操作介面

我們為 Wedjat 設計的使用者介面如圖 22 所示，第一頁有三個按鍵，讓使用者可以選擇設定自己的喜好，裝載藥物處方籤檔案，顯示目前的排程結果。在 Preference Setting 的頁面，使用者可以設定自己的服藥習慣，作息，以及某些不想被排程的特殊時段。使用者設定完自己的喜好與裝載自己的處方籤後，可選擇啟動排程。排程後的結果會顯示在 Show Schedule 的頁面，同時也會寫入 Calendar Application 裡，使用者可以在這兩個地方查詢自己的的服藥時程。

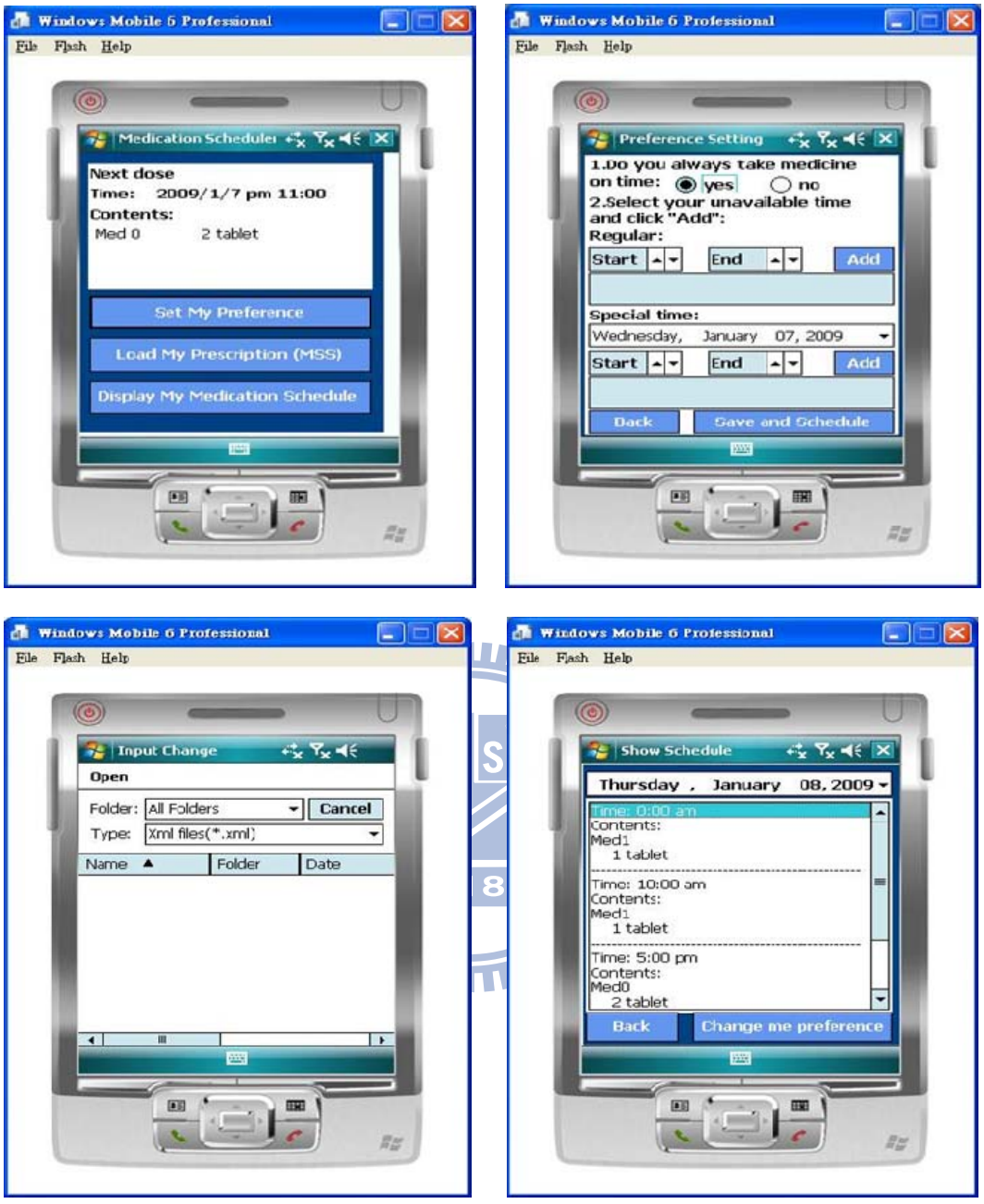


圖 22: Wedjat 使用者介面

第 6 章　結語

6.1　研究成果

本研究的主要貢獻有下列點：

- 設計依不同需求與重要性的劑量分群服用的方法
- 藥物時程的劑量服用彈性方法的改善
- 實作 Wedjat 的原型，OMAT 和 ODAT 演算法
- 將 Wedjat 與行動手機上的行事曆提醒功能結合，由行事曆主動發出用藥提醒

本研究中仍然少數的設計尚未實作完成，如下所示：

- 劑量的分群方法
- 藥物資訊 MSS 與資料庫的整合

6.2　未來方向

本研究未來的可行方向，能使本研究更趨完整，敘述如下：

- Wedja 與遠端個人健康紀錄系統整合，以達到遠距監控的目的
- 與其他個人健康管理服務整合，像是 Biofeedback 感測器等等

參考文獻

[1] Kohn LT, Corrigan JM, Donaldson MS (Ed.), *To Err Is Human: Building a Safer Health System*. Compiled by the Committee on Quality of Health Care in America, the Institute of Medicine, US National Academies; published by National Academy Press, 1999.

[2] Wan D, "Magic Medicine Cabinet: A Situated Portal for Consumer Healthcare," *Proceedings of 1st International Symposium on Handheld and Ubiquitous Computing (HUC '99)*, September 1999.

[3] Murray MD, "Automated medication dispensing devices," Chapter 11 in *Making Healthcare Safer: a Critical Analysis of Patient Safety*, 01-E58, Agent for Healthcare Research and Quality, 2001.

[4] Governo M, Riva V, Fiorini P, Nugent C, "MEDICATE Tele-assistance System", *11th International Conference on Advance Robotics*, June 2003.

[5] Liu JWS, Shih CS, Tsai PH, Yeh HC, Hsiu PC, Yu CY, Chang WH, "End-User Support for Error Free Medication Process," Proceedings of High-Confidence Medication Device Software and Systems, pp. 34 – 45, June 2007.

[6] P. C. Hsiu, H. C. Yeh, P. H. Tsai, C. S. Shih, D. H. Burkhardt, T. W. Kuo, J. W. S. Liu and T. Y. Huang, "A General Model for Medication Scheduling," Institute of Information Sciences, Academia Sinica, Taiwan, Technical Report No. TR-IIS-05-008, October 2005.

[7] PDRHealth: http://www.pdrhealth.com/drugs/

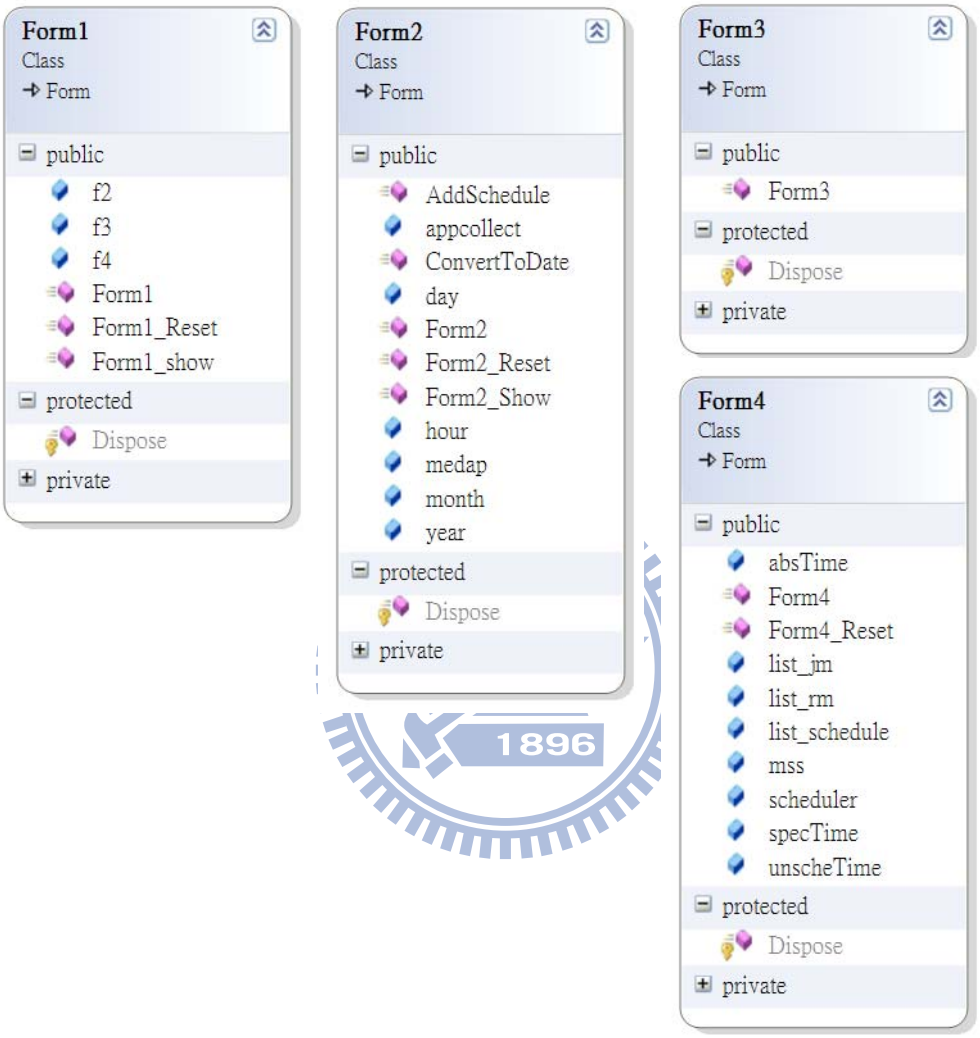
[8] Tsai PH, Shih CS, and Liu JWS, "Algorithms for scheduling multiple interacting medications," *Technical Report TR-IIS-08-001*, Institute of Information Science Academia Sinica, Taiwan, April 2008.

[9] Jane W.S. Liu, Real-Time Systems, Prentice-Hall, America, 2000.

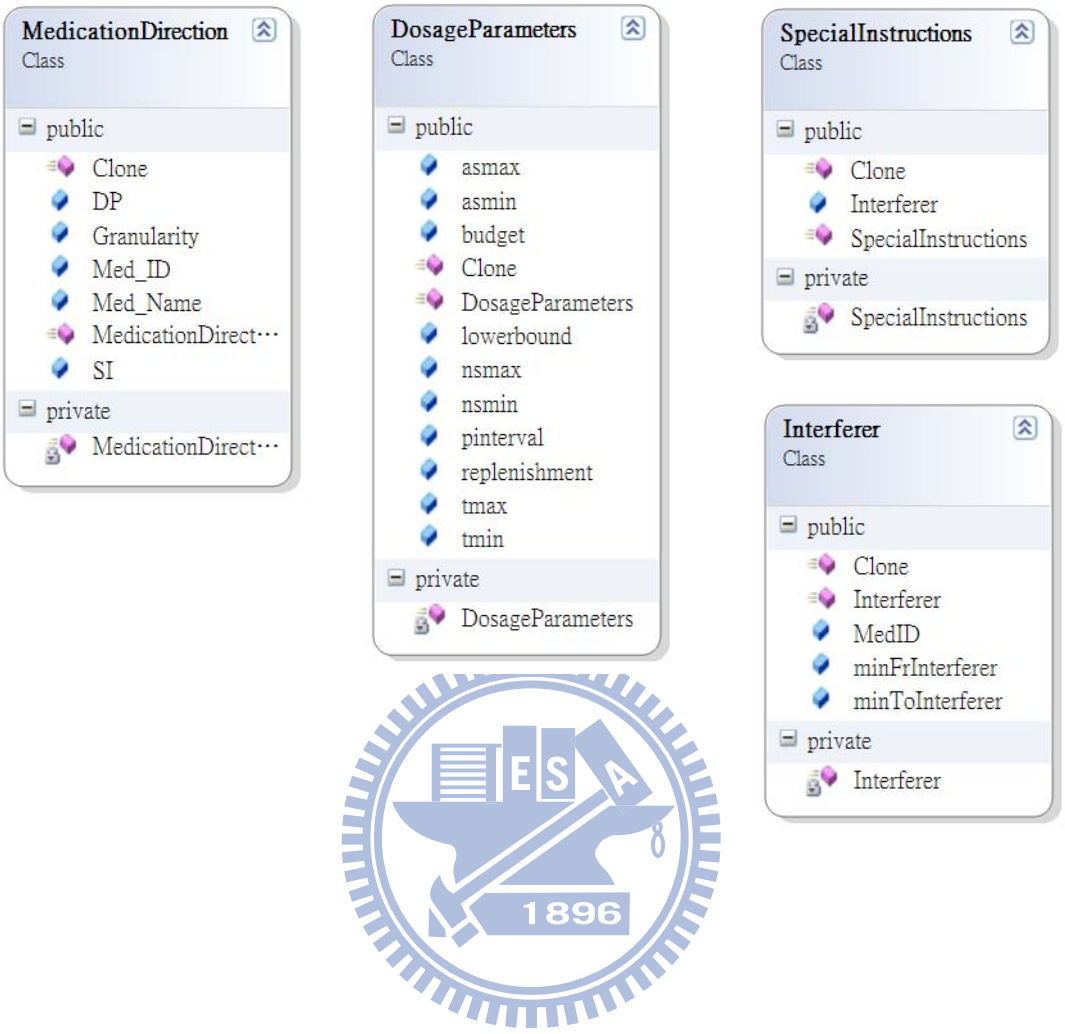
[10] Evangelos Triantaphyllou, Multi-Criteria Decision Making Method: A Comparative Study, Kluwer Academic Publishers, America, 2000.

附錄

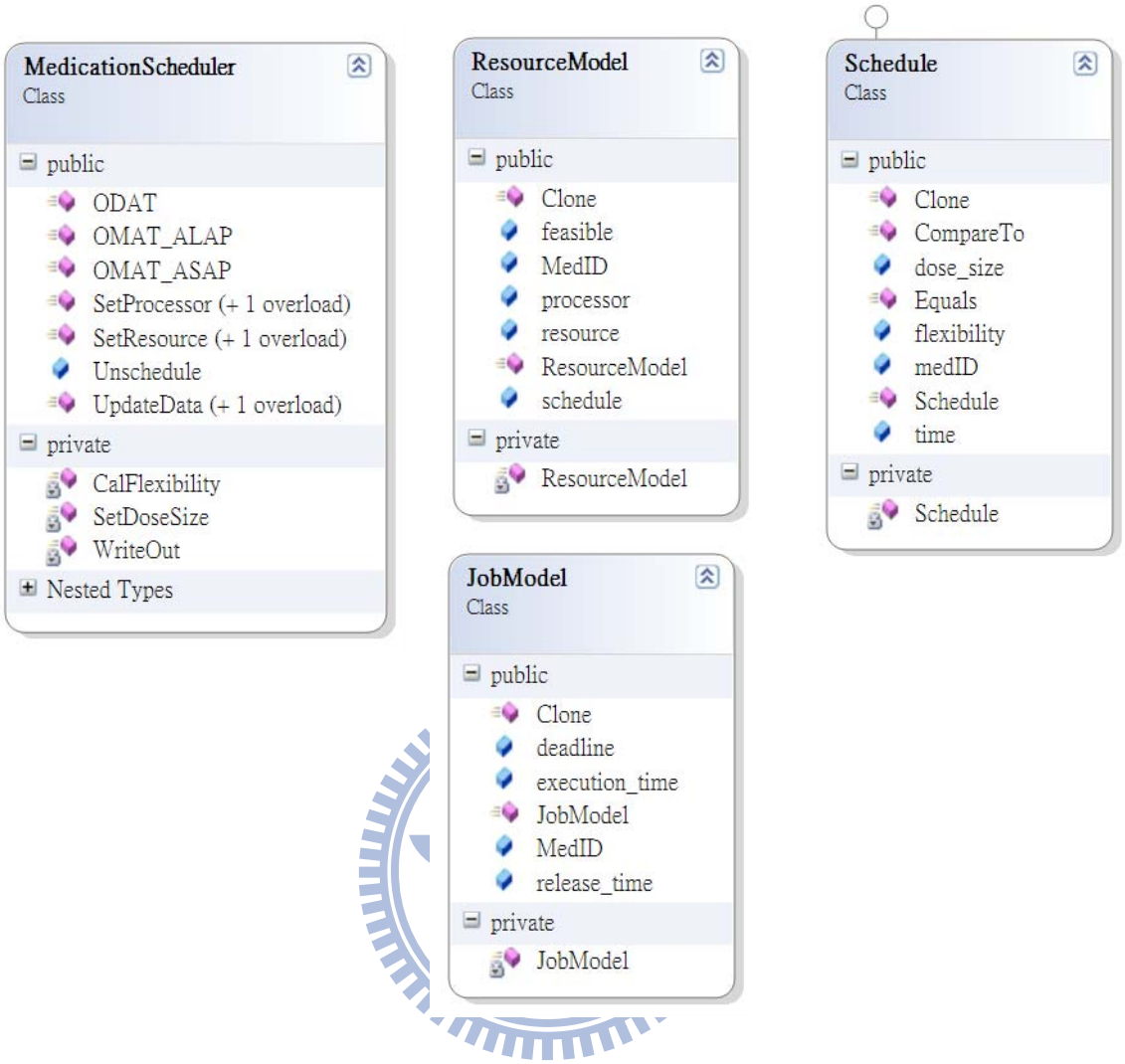
使用者頁面的 Class Diagrams：



藥物資訊的資料結構：


MedicationDirection
Class
public
Clone
DP
Granularity
Med_ID
Med_Name
MedicationDirect…
SI
private
MedicationDirect…
DosageParameters
Class
public
asmax
asmin
budget
Clone
DosageParameters
lowerbound
nsmax
nsmin
pinterval
replenishment
tmax
tmin
private
DosageParameters
SpecialInstructions
Class
public
Clone
Interferer
SpecialInstructions
private
SpecialInstructions
Interferer
Class
public
Clone
Interferer
MedID
minFrInterferer
minToInterferer
private
Interferer

藥物排程器相關的 Class Diagrams：



藥物優先順序法相關之 Class Diagrams：