中央研究院
資訊科學研究所
Institute of Information Science, Academia Sinica • Taipei, Taiwan, ROC

TR-IIS-05-008

A General Model for Medication Scheduling

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

October 2005 || Technical Report No. TR-IIS-05-008

http://www.iis.sinica.edu.tw/LIB/TechReport/tr2005/tr05.html

Institute of Information Science, Academia Sinica

Technical Report TR-IIS-05-008

A General Model for Medication Scheduling

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*

Abstract

We describe here a general model, called the GMS (General Medication Schedule) model, for specifying requirements and constraints of medication schedules. A specification based on the model captures limits in dosage and timing given by medication directions in general and guidelines and preferences specific to the user. The controller of a smart dispenser can compute schedules of dispenser commands based on the specification. By executing the commands according to the schedules, the dispenser deliveries reminders and dispenses medications in conformance with directions. The rules embedded in the specification provide the dispenser with criteria needed for compliance monitoring and enforcement.

---

* J. W. S. Liu is affiliated with Institute of Information Science, Academia Sinica, Taiwan. P. C. Hsiu, H. C. Yeh, C. S. Shih and T. W. Kuo are affiliated with Department of Computer Science and Information Science, National Taiwan University, Taiwan. P. H. Tsai and T. Y. Huang are affiliated with Department of Computer Science, National Tsing Hua University, Taiwan. D. H. Burkhardt is affiliated with School of Medicine, University of California at San Francisco, USA.

Table of Contents

1 Introduction

These days, searches over the web for medication schedules usually bring up numerous papers and sites on medication compliance, too numerous to be cited here. For a person on medications that are free of undesirable interactions, compliance means taking the correct dosage of each at the right times, in the correct manner and for the prescribed duration. The importance of compliance cannot be exaggerated for the elderly. Many modern drugs can do wonders in combating aging and chronic illnesses, but only if one follows the prescribed course of treatment. Unfortunately, non-compliance is far too common and severe[1]. An elderly individual may take four to five prescribed medications at a time, in addition to some over-the-counter drugs and supplements. The person may have over 10 different prescriptions each year. Following all the directions consistently day in day out, for months and years, is demanding even for someone who is still healthful and energetic. It is even more so for someone who has become increasingly more forgetful and easily fatigued from activities of daily living.

Many devices and services are now available to ease the effort and improve the chance in compliance. They include pillboxes and medicine dispensers (e.g., [1, 2]), as well as websites that help the user to generate and format schedules of one or more medications (e.g., [3]). In this report, we confine our attention to smart dispensers. A smart *medication dispenser* reminds its user at times when medications should be taken, controls their dosages each time, monitors user actions, provides the user with warnings or notifies a caretaker when instances of non-compliance are detected. For the dispenser to be effective, it must send reminders and dispense medications correctly. Being correct is insufficient, however. Being user friendly is also important. In particular, it must not be overly rigid: The schedules it uses should provide the user with flexibility whenever possible and be easy to follow as much as possible. It must not be an alarmist. False alarms are unavoidable, but they should occur infrequently.

This report describes a general model that captures limits and constraints given by medication directions prescribed for the user and guidelines and preferences provided by the user. The model, called the GMS (General Medication Schedule) model, is a rigorous foundation for specifying medication schedules and reasoning about their correctness. The controller of a smart dispenser can compute schedules of dispenser commands based on a GMS specification. By executing the

[1] According to the March 2005 survey in http://www.harrisinteractive.com/news/allnewsbydate.asp?NewsID=904, one of three US adults who regularly takes prescription drugs reports often and very often non-compliance.

commands according to the schedule, the dispenser deliveries reminders and dispenses correct doses of medications at correct times. Furthermore, the GMS specification provides the dispenser with criteria needed for compliance monitoring and enforcement.

Key Assumptions and Definitions We focus here on the problem of how to specify for the smart dispenser of a single user, the requirements and constraints of medication schedules. The *core part* of the specification is derived from the directions of all the medications taken by the user. The core may be augmented by user preferences, habits and behavior. The requirements and constraints specified by the core part are mandatory. They take precedence over the *augmented part* of the specification.

Specifically, we assume that the core part of the GMS specification for the user is generated and downloaded to the dispenser by an external tool (or manually by a physcian, pharmacist or other health-care professional with the aid of a tool) via the Internet or some form of removable storage device. We call such a tool an *authoring tool* and will describe it in a separate report. Similarly, updates in directions and hence the core part of the GMS specification used by the dispenser are generated by the tool (or manually) and downloaded to the dispenser.

The authoring tool has access to drug libraries and on-line information and can derive from them general directions of all the medications managed by the dispenser. In addition, it has user's prescriptions containing directions specific to the user. In the process of deriving the core part of the GMS specification, the tool has already verified automatically or interactively the following facts:

1. It is safe for the user to take all the medications for the prescribed duration.
2. The medication directions given to the user (and dispenser) have already been adjusted to take into account the effects of interactions among the medications and food.

In contrast, a dispenser has no access to information in addition to a GMS specification. In particular, the dispenser has no access to drug libraries and on-line medication advisories and contains no cached information sufficient for dynamic verification purposes. The dispenser may store the user's prescriptions as records, but has no capability to extract directions from them.

The augmented part of the GMS specification is generated and maintained locally by the dispenser. We will return in Section 4 to describe the kind of user interface a dispenser may use to capture user provided data on preferences and habits. With the user's permission, a smart dispenser can record statistics on these parameters in order to better serve the user. Subsequent discussions assume that the dispenser has the user’s permission and is configured to monitor and record medication history and user behavior.

We consider here solely of medications that are dispensed and taken in discrete units. In other words, our discussions do not apply to medications that are delivered in continuous drips. We use the term *dispensing job* (or simply a *job*) of a medication to mean the following sequence of actions by the dispenser to administer a dose of the medication:

1. Send a reminder to tell the user it is time to take the medication,
2. Wait and monitor until the user comes and retrieves the medication,
3. Mark the time of the retrieval,
4. Check for compliance and take specified action(s) when non-compliance is detected,
5. Record the time of retrieval, (user) promptness and other statistics for later use.

By *promptness*, we mean the amount of time to complete Step 2: It is the length of time the user takes to come and retrieve a medication after being reminded. The time instants when Step 1 starts and Step 3 completes are the *start time* and *completion time* of the job, respectively. The total amount of time between the start time and completion time of a job is its *elapse time*. In definitions presented in subsequent sections, by a *time instant when a medication is administered* or *dispensed*, we mean the completion time of the associated dispensing job.

An assumption here is that the amounts of time for the dispenser to carry out the first and third steps are negligibly small compared with the elapse time of the job. This assumption is valid when the dispenser takes 1~100 milliseconds to do them. Another assumption is that one or more dispensing jobs can start while the dispenser is carrying out Steps 4 and 5 of a previous job. This assumption is valid for a multi-task dispenser that has sufficient processing, communication and storage resources to do its work on a timely basis.

Related Work The dispenser scheduler we are developing takes as input interface, command and functional specifications of the dispenser, together with the GMS specification for the user. In this way, the scheduler can generate schedules of reminders, commands, notifications for any dispenser, including the smart ones already on the market (e.g., [1, 2]). The GMS model is not intended for web-based scheduling tools, such as the one at [3], however. Those tools have, or should have, on-line access to the most up-to-date directions and alerts.

The problem addressed here is much narrower than the problems that must be dealt with by projects on on-line medication consultations. An example is the medication advisor in [4]. The intelligent tool converses with the user in a natural language, tries to understand meanings and intentions of the user and medication directions in drug libraries, and offers to the user advices and information on any combination of prescription and over-the-counter medications. Most of the difficult problems they address are out of scope for us. On the other hand, their treatment to medication scheduling does not reach sufficient depth and rigor for our purpose.

In many aspects, the GMS model extends well-known models (e.g., [5-10]) developed for specifying timing requirements of real-time computation and communication tasks. This is why some of the GMS model parameters have identical or similar names as parameters of some real-time models. Rather than enumerating here the differences between the GMS model and existing real-time models, we will discuss specific relationships between them in subsequent sections where GMS model elements are introduced.

Organization of the Report In the remainder of the report, we first confine our attention to directions of individual medications, assuming that the user takes only one medication. We then return to the general case where a user may be on multiple medications.

Specifically, Sections 2 and 3 focus the core part of the GMS specification. Section 2 describes model elements derived from the direction of a single medication. Section 3 describes model elements that specify interactions among medications and additional constraints arisen from the interactions. Section 4 discusses the augmented part: parameters that capture the preferences and habits of the user. Section 5 describes a graphical representation that facilitates the validation of limits and constraints of medication schedule(s). Section 6 summarizes our current work and future plan. The appendix lists terms and notations used in the report.

2 Direction Parameters

The direction of a medication specifies how the medication is to be dispensed and what compliance means. As an example, we look at the direction for an over-the-counter pain killer:

> ***Example 1***: "1 gel caplet every 4 to 6 hours while symptom persists. If pain or fever does not respond to 1 caplet, 2 caplets may be used but do not exceed 6 gel caplets in 24 hours unless directed by a doctor. The smallest effective dose should be used. Take with food or milk if mild heartburn or upset stomach occurs while taking the medication."

The direction specifies the following parameters:

1. The nominal minimum dose (1 caplet) and nominal maximum dose (2 caplets) taken each time,
2. The minimum and maximum lengths of time (4 to 6 hours) between consecutive doses,
3. A limit on the total intake (at most 6 caplets) in a given time interval (24 hours),
4. Relationship with food (can be taken with food), and
5. A condition of optimality or figure of merit (the smallest effective dose is optimal).

Generalizing from this example, we express the direction for a medication *M* more formally in terms of *direction parameters*, including the ones defined in Figure 1.

M (g, [T_{min}, T_{max}], [d_{min}, d_{max}], [s_{min}, s_{max}], (B, R), (L, P), F)

- M: Name of the medication
- g: Granularity of dosage
- [T_{min}, T_{max}]: Minimum and maximum durations over which the medication is administered
- [d_{min}, d_{max}]: Nominal minimum and maximum sizes of dose taken each time
- [D_{min}, D_{max}]: Absolute minimum and maximum sizes of dose taken each time
- [s_{min}, s_{max}]: Nominal minimum and maximum temporal separations between (consecutive) times when the medication is taken
- [S_{min}, S_{max}]: Absolute minimum and maximum temporal separations
- (B, R): Maximum overall intake over a specified time interval given by budget B and replenishment delay R
- (L, P): Minimum overall intake over a specified time interval given by lower threshold L and a period length P
- F: Relationship with food
- Non-compliance event types and corresponding actions [11]

Figure 1 Direction Parameters

We define in this section all the parameters listed in Figure 1, except the last two. The flag F indicates whether the medication is taken before, with or after food and drink, or can be taken independent of food. We will return to address this relationship in Section 3. What non-compliance means and how serious it is are medication specific. This is why Figure 1 lists non-compliance event types and corresponding actions by the dispenser as direction parameters. This section defines several non-compliance event types involving only one medication. Section 3 will discuss non-compliance that may arise from interactions among multiple medications. We defer the descriptions of dispenser actions to a separate report [11] on interfaces and designs of medication dispensers.

There may be additional parameters, and some of the parameters listed in Figure 1 may have additional elements. Typically, normal operations of a smart dispenser are guided and timed by nominal values of direction parameters and overall limits. The dispenser treats absolute values as hard constraints and uses them to support its compliance monitoring and notification decisions. In examples throughout the report, we omit absolute values in direction parameter lists.

2.1 Granularity and Duration

As stated earlier, all medications considered here are dispensed in discrete units. The *granularity* of the units is denoted by g: A granule may be a tablet or caplet, or number of milligrams (mg), number of cubic centimeters (cc) and so on. For an individual medication, it is convenient to

make the granularity equal to the size of the minimal dose that can be administered. So, in Example 1, g is one caplet. The granularity of baby aspirin taken one per day by the elderly for prevention of heart attack may be stated as 81 milligrams or one baby aspirin tablet.

$[T_{min}, T_{max}]$ gives the range of the duration, in length of time or total number of doses, over which the medication is to be administered. We call T_{min} and T_{max} the *minimum duration* and the *maximum duration*, respectively, of the medication direction and say that the direction has a *fixed duration* when T_{min} and T_{max} are equal.

Generally speaking, T_{min} is larger than zero for a medication (e.g. antibiotics) that must be taken for at least some prescribed length of time. T_{max} is finite for any medication that should not be taken beyond some length of time for reasons such as habit forming or undesirable cumulative side effect. When used to administer a medication following a direction with minimum duration longer than how long the supply lasts, a smart dispenser should warn the user or caretaker when supply is about to run out, send notification if supply is not replenished in time, and follow the direction in dispensing and compliance monitoring across batches of supply.

The duration may be unconstrained (i.e., $[T_{min}, T_{max}]$ is equal to $[0, \infty]$, where ∞ denotes infinity) as would be the case of a medication that is taken as needed and posts no danger in long term use. Both T_{min} and T_{max} may be infinite. An infinite duration means that the administration of the medication is to continue until terminated explicitly. This is the case of baby aspirin for the elderly, as well as medications for chronic conditions.

2.2 Dose Size and Separation

Dose size is the number of granules taken each time. A smart dispenser normally provides the user with doses of sizes in the range $[d_{min}, d_{max}]$. We call the lower limit d_{min} and upper limit d_{max} *nominal minimum dose* and *nominal maximum dose*, respectively. They are the values stated explicitly in the direction. In Example 1, the values are 1 and 2 caplets, respectively.

The *temporal separation* (or *separation* for short) of a medication is the length of time between consecutive instants when the medication is administered or dispensed. Temporal separation is within in the range $[s_{min}, s_{max}]$, where s_{min} and s_{max} are the *nominal minimum separation* and *nominal maximum separation,* respectively. The dispenser is scheduled to administer doses separated in time within the nominal range.

The combination of minimum dose and maximum separation defines the *smallest dosage*. Similarly, the combination of maximum dose and minimum separation defines the *largest dosage*.

In addition to their nominal values, dose size and separation may have absolute values that differ from the corresponding nominal values. We denote the absolute minimum dose and the absolute maximum separation by D_{min} *and* S_{max}, respectively. The *absolute minimum dose* D_{min} is the smallest dose size amongst all values that are explicitly or implicitly specified by statements in the direction. Similarly, the *absolute maximum separation* S_{max} is the largest value specified by statements in the direction and, hence, may be larger than the nominal maximum separation. To explain, let us look at Example 1 again. In addition to the first point in the list above, the 5th point is also about dose size and separation. The optimality condition says that the user does not have to take the pain killer. A dispenser with compliance monitoring capability should not sound alarm if the user skips some doses. Hence, the minimum dose is not 1 caplet; it is actually 0. The absolute maximum separation is infinity.

The *absolute maximum dose* D_{max} can be defined analogously: It is the largest upper limit given by statements in the general direction, if no specific limit is prescribed for the user. Otherwise, it is the value prescribed for the user. For the pain killer in Example 1, the absolute maximum dose and nominal maximum dose are the same. On the other hand, the direction of a calcium supplement stating "take one tablet twice daily with food" gives a nominal maximum dose of one tablet. While not the most desirable, it is safe for the user to take two tablets together once a day (or even two tablets twice a day), especially if this is done infrequently. So, the absolute maximum dose can be set at two tablets.

Similarly, the absolute shortest time between two doses may actually be smaller than the nominal value stated explicitly by the direction. We call the absolute limit *absolute minimum separation* $S_{min.}$ For Example 1, the absolute minimum separation is equal to the nominal minimum separation of 4 hours.

As another example, below are the statements about separation in the direction of Lanoxin, a brand of digoxin [12]. (Digoxin is used for treatment of some serious heart problems.)

> ***Example 2***: "Lanoxin usually is taken once daily. To help you remember your dose, try to take it at the same time every day … It's best to take this medicine on an empty stomach. However, if this upsets your stomach, you can take Lanoxin with food. If you miss a dose … If you remember within 12 hours, take it immediately. If you remember later, skip the dose you missed and go back to your regular schedule. Never take 2 doses at the same time. If you miss doses 2 or more days in a row, consult your doctor."

In this case, the nominal minimum and maximum separations are both 24 hours. The absolute minimum separation is 12+ hours, and the absolute maximum separation is 36- hours. We have

these separations between three doses when the user takes the first and third doses on time and the second one at the latest allowed time (i.e., the first one at t, the second at t + 36-, and the third one at t + 48).

2.3 Maximum Total Intake Constraint

Two limits govern the total number of granules taken over a specified time interval: the maximum number of granules the user is allowed to take and the minimum number of granules the user must take. They are called the *maximum total intake* and *minimum total intake* (or *maximum intake* and *minimum intake* for short), respectively.

Budget and Replenishment Delay We specify the maximum overall intake by a 2-tuple (B, R): No more than B granules of M can be taken in any time interval of length R. B stands for *budget*, and R stands for *replenishment delay*. When a dose of size d is dispensed at time t, d granules of budget are consumed, and the d-granule chunk will be replenished at time $t + R$. At any time, the user can take no more of the medication than there is budget for it.

A smart dispenser can monitor the total intake and enforce the constraint of a medication with maximum intake (B, R) in the following manner: It maintains the current budget b at all time. At the start, before any of the medication is ever dispensed, b is equal to budget B. When it dispenses a dose of size d at time t, the dispenser decrements the current budget b by d and sets time for replenishing the chunk of budget at time $t + R$. At time $t + R$, it replenishes the chunk. The dispenser stops dispensing as long as the current budget b is insufficient to meet the demand of dose size d. This way of keeping track of the overall limit is similar in principle to how a sporadic server limits processor bandwidth consumption of aperiodic tasks [6].

Figure 2 illustrates how this scheme would work for the pain killer in Example 1. The budget and replenishment delay are 6 caplets and 24 hours. The numbers below the horizontal axis are hours from the time the user purchased the medication. The vertical axis is labeled by number of caplets. The dotted line plots the amount of budget remaining as a function of time. The heavy solid line plots the total intake within the past 24 hours. The initial value of the budget b is 6.

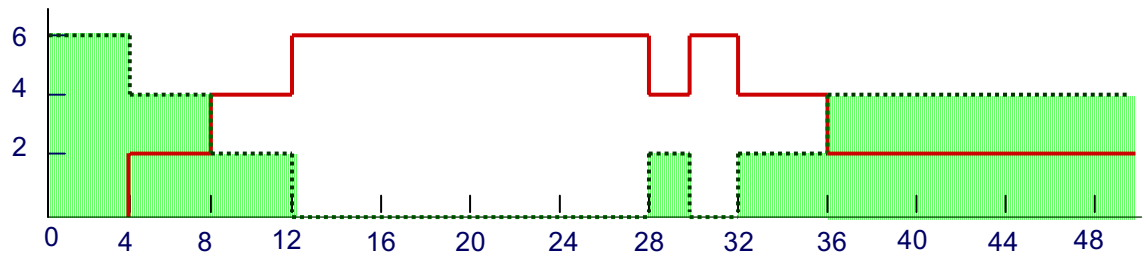

Figure 2 Example on Maintaining Maximum Intake Constraint

Suppose that the user asks for 2 caplets at time 4, 8 and 12 (hours). Each time, the dispenser gives the user the requested dose and decrements the budget by 2. At time 12, there is no more budget. The dispenser would refuse to dispense any caplet if it were asked to do so. The 2-caplet budget decremented at time 4 is replenished at time 28. Suppose that the user does not ask for more pain killer at that time. Then the budget becomes 2. When the user asks for the medication at time 30, the dispenser grants the request, exhausting the budget again. The figure shows the case where the user asks no more medication after time 30. The budget increases by 2 at time 32 and 36 and becomes 4 at 36. It remains at 4 until time 54 and becomes 6 at that time because the 2-caplet budget consumed at 30 will be replenished then. We see that by keeping track of the budget in manner described above, the dispenser ensures that the intake of the user is within the specified limit.

Hard Vs Soft Constraints The maximum intake as defined above is a *hard* constraint, in the sense that the dose dispensed at any time should never exceed the current available budget. Some medication may have a *soft* maximum intake, which may be exceeded occasionally; this is a way to provide some flexibility in scheduling in a safe way. A soft maximum intake constraint may be specified in a number of ways. As an example, it can be defined by a 3-tuple (*B, E, R*). The meaning of *R* remains the same. The budget *B* is soft, meaning that the total intake over any interval of length *R* should be no more than *B* most of time, but an occasional violation of the limit is acceptable. The direction also specifies an *excess E*: $B + E$ defines a hard limit that should never be exceeded. Then the goal of scheduling is to minimize the number of times (or the average fraction of time) the total intake over intervals of length *R* exceeds *B*, or to make sure that in every *m* (>0) intervals of length *R*, the total intake exceeds the soft limit *B* no more than *k* (≥0) times, and so on. These goals resemble (or are analogous to) the goals of soft real-time scheduling, such as to minimize miss rate, to minimize the average excess, to meet (*m, k*)-firm deadline and so on [5-8].

Alternatively, rather than a fixed replenishment delay *R*, the medication may allow an advance *A* (≥0) in delay. In other words, the budget consumed at *t* can be replenished in the interval ($R - A, R$). Indeed, almost every kind of soft timing constraint for real-time systems and applications can be modified and applied to soft limits on the total medication intake.

Example 3: We can treat food as a special kind of medication. Its budget limit *B* is 100, meaning a nominal 100 % of food consumed per day by the user. Its replenishment delay is 24 hours. The direction for food intake of a user who usually eats three meals and a snack at 8 AM, 12 noon, 6 PM and 8 PM may be written as

Food (1, [∞, ∞], [10, 40], [2, 12], (100, 50, 24), (50, 24)-*Periodic*, TRUE)

Here, the maximum intake is a soft constraint with an allowed excess of 50 %. Indeed for this "medication", soft maximum intake is more appropriate. While it is desirable not to over eat, everyone wants to and can do now and then.

Hereafter, we will concentrate primarily on the case of hard constraints. Except for food, by maximum intake, we mean hard maximum total intake constraint.

2.4 Minimum Total Intake Constraint

We denote the minimum overall intake by the 2-tuple (*L, P*), where *L* specifies the minimum number of granules and *P* specifies the length of a time interval. There are two kinds of *minimum* (total) *intake constraint*: (*L, P*)-*Periodic* and (*L, P*)-*Uniform*.

For a medication with (*L, P*)-*Periodic* minimum intake constraint, time is segmented into periods of length *P* starting from an initial phase (e.g., the time instant when the medication is taken for the first time). *L* granules of a medication with (*L, P*)-*Periodic* minimum intake constraint must be taken in every period. The constraint is named periodic because it is analogous to a mandatory periodic task in the imprecise computation model [7]: A mandatory periodic task (*L, P*) must execute for at least *L* units of time in every period of length *P*.

The constraint (*L, P*)-*Uniform* is analogous to maximum intake (*B, R*). Minimum intake (*L, P*)-*Uniform* requires that the total number of granules taken within any interval of length *P* must be at least equal to *L*.

Figure 3 illustrates the difference between (*L, P*)-*Periodic* and (*L, P*)-*Uniform* constraints: The numbers on the time line indicate time in hours elapsed since the time origin 0; say time 0 is the start of day 1 when the user begins to take medications. The figure shows schedules of two medications. The positions of the tablets and caplets indicate when they are taken.

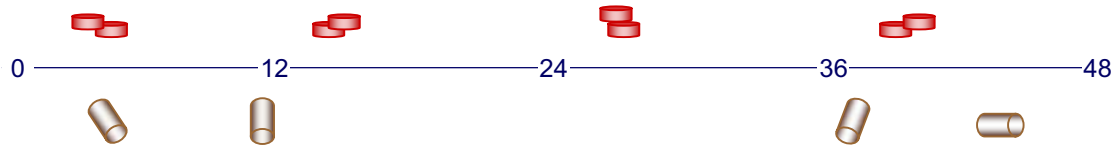

Figure 3 Two Kinds of Minimum Intake Constraints

From the schedule of tablets above the time line in Figure 3, we can see that if we slide a 24-hour window across the time line, there are always 4 tablets in the window. Hence, the schedule meets the (4, 24)-*Uniform* constraint. It also meets the (4, 24)-*Periodic* constraint since there are 4 tablets in each of the periods (0, 24] and (24, 48].

The caplets under the time line may be a calcium supplement, which the user takes following the direction below:

Example 4: Direction for calcium supplement reads "Take one tablet with food twice a day". The direction parameters can be written as

Calcium (1, [∞, ∞], [1, 2], [0, 24], (4, 24), (2, 24)-*Periodic*, TRUE)

The positions of the caplets in Figure 3 indicate that they are taken on the first day together with breakfast and an early lunch and on the second day with a late lunch and an evening snack. To determine whether this schedule meets the (2, 24)-*Periodic* constraint, we divide time into 24-hour periods: (0, 24] and (24, 48]. The schedule meets the constraint because two caplets are taken within each of the two periods. The schedule does not meet the (2, 24)-*Uniform* constraint because no tablets are taken in the 24 hour interval (12, 36)[2]. (2, 24)-*Uniform* is too stringent for calcium supplement, however, since going on 24-hours without calcium pills does not matter.

Returning to Example 2, the user must not miss doses 2 or more days in a row. This requirement can be stated as (1, 60)-*Uniform* constraint. (Two doses in a row are missed if a user misses a dose on a day and then fails to take a dose 36 or more hours later.) Below is another example:

***Example 5*:** The direction for penicillin when used to treat of certain bacteria infection reads "Take 250 to 500 milligrams (mg) every eight hours. Keep taking this medicine for at least ten days even if you begin to feel better after a few days. Do not miss any doses. Also, it is best to take the doses at evenly spaced times, day and night with 8 or more ounces of water on an empty stomach (either 1 hour before or 2 hours after meals)." In our notation, the direction says

Penicillin (1 mg, [240, 240], [250, 500], [8, 8], (1500, 24), (750, 24)-*Uniform*, TRUE)

The unit of time used here is hour.

The constraint (*L, P*)-*Uniform* is more appropriate for drugs such as digoxin and antibiotics. It is important that a certain amount of such a medication is at work at all times. However, it is not appropriate for many medications taken for health maintenance purpose over long periods of time. For most medications, the number of doses in a constraint interval is small. The more stringent constraint (*L, P*)-*Uniform* invariably forces the doses be scheduled evenly spaced in time. Such schedules would be overly rigid, hard to follow.

---

[2] We note that if the user were to repeat the first-day schedule on the third day, 4 tablets would be taken in the 24 hours [36, 60]. This schedule is allowed because the maximum intake constraint is (4, 24).

2.5 Time Varying Direction

Thus far, we considered only static directions: A *static direction* is specified by parameters that have constant values. In general, the direction of a medication may change with time. As an example, we consider a user whose doctor puts him/her on Diabinese, generically known as chlorpropamide, when they discover that the user has severe type-2 diabetes.

> ***Example 6:*** Dosage information posted at [12] states "Usually, an initial daily dose of 250 milligrams is recommended for stable, middle-aged, non-insulin-dependent diabetics. After 5 to 7 days, your doctor may adjust this dosage in increments of 50 to 125 milligrams every 3 to 5 days to achieve the best benefit. People with mild diabetes may respond well to daily doses of 100 milligrams or less of Diabinese, while those with severe diabetes may require 500 milligrams daily."

Suppose that the doctor tells the user to take 250 milligrams each day for 7 days and then increase the dose by 50 milligrams every 3 days until the dose size reaches 500 milligrams daily. Hereafter, the user is to take 500 milligram daily until instructed to do otherwise. Although seemingly complicated, the user can easily follow such a direction with the help of a smart dispenser. Clearly, we need to expand the set of parameters in Figure 1 in order to capture the time varying nature of the direction.

A natural way is to treat a time varying direction as a sequence of static directions. Formally, we write a time varying direction $\boldsymbol{M}(N_M)$ that changes N_M-1 times as an indexed set of N_M static directions:

$$\boldsymbol{M}(N_M) = \{M_1, M_2, \ldots M_N\}$$

Each static direction M_i is defined by constant parameters, including the ones discussed above. The dispenser uses the static directions in ascending order of their indices. In other words, the end of M_i is the beginning of M_{i+1} for i = 1, 2, ⋯, N_M-1. Except for the last direction M_N, all static directions in the sequence have fixed durations. The duration of the time varying direction is the sum of the durations of the static directions contained in it. The dispenser is required to monitor and enforce separation and intake constraints across of the static directions.

Returning to the previous example, we see that the time varying direction for the user on Diabinese is a sequence of six static directions: The first one is for duration of 7 days with a daily dosage of 250 milligrams, the next four are for duration of 3 days each and daily dosages of 300, 350, 400, and 450 milligrams, and the sixth one specifies a daily dosage of 500 milligrams for an infinite duration.

2.6 Some Non-Compliance Event Types

Absolute dose and separation parameters and intake constraints of a medication allow us to define several non-compliance event types: They are under-dose, under-medicate, over-dose, over-medicate and concurrent event types. Non-compliance events of these types involve only one medication. Section 3 will define non-compliance types arisen from multiple medications.

The dispenser treats an event of the user taking a dose of size less than the absolute minimum dose size D_{min}, or two consecutive doses further apart than the absolute maximum separation S_{max}, or both as an *under-dose event*: Such an event arises from non-compliant to direction for a single dose. An *under-medicate event* occurs when user's total intake over a specified interval is observed to fall below the specified lower limit; in other words, the minimum intake constraint is violated. Taking a medication for a shorter time than its minimum duration is also an under-medicate event.

Similarly, an *overdose event* occurs when the user takes a dose of size larger than the absolute maximum dose size, or two doses separated in time less than the absolute minimum separation, or both. A violation of the maximum intake constraint is an *over-medicate* event.

A medication direction may include statements on that two or more actions must be done (or must not be done) at the same time. Examples of such statements are "Never take 2 doses at the same time" in Example 2 and "take the doses … on empty stomach …" (i.e., do not take the medication and food at the same time) in Example 4. We call a constraint imposed by such a statement a *concurrency constraint,* and a (*non-compliance*) *concurrent event* is an occurrence of non-compliance to a concurrency constraint. Many concurrency constraints are, however, subsumed by constraints on dosage and intake limits. The prohibition of taking multiple doses at the same time is often less stringent than the absolute minimum separation constraint. The constraint of "do not take the medication at the same time with food" is equivalent to the more general constraint on the minimum separation between each dose and food. We will return to elaborate this kind of constraint further in Section 3.

We use the term *non-compliance event* to mean an event of any of the types defined above and in Section 3. When a non-compliance event occurs, the dispenser records the occurrence and takes appropriate action(s). The action may be merely to record the occurrence, or alert the user or a caretaker, and so on. What action should be taken depends on the event type, the medication and the user. In the GMS model, the types of non-compliance events and the corresponding actions of the dispenser are direction parameters.

3 Dependencies of Multiple Medications

In general, a user may take more than one medication. Someone or some tool must verify that the user can safely take all of them. Medication dispensers are unlikely to have this capability. Hence, we assume in this report the fact that the user can safely take all the medications managed by a dispenser has already been verified. Moreover, information on how to modify directions of the medications in order to keep the effects of interaction tolerable is provided along with the directions. Our question is simply how to define for the dispenser the additional scheduling constraints that are imposed for the sake of ensuring safe interaction.

To provide rationale, we look at the following examples: Suppose that an elderly user takes aspirin, Fosamax, Lipitor and Mylanta. The generic names for Fosamax and Liptor are alendronate and atorvastatin, respectively, while Mylanta tablets contain calcium carbonate and magnesium hydroxide.

> ***Example 7:*** Information on possible interactions between Fosamax and drugs and food [12] states "Combining aspirin with a Fosamax dose of more than 10 milligrams per day will increase the likelihood of stomach upset. Calcium supplements such as Caltrate, antacids such as Riopan, and some other oral medications will interfere with the absorption of Fosamax, so wait at least 30 minutes after taking Fosamax before you take anything else."
>
> ***Example 8:*** A user may take Mylanta occasionally for relieve from acid reflux burn. Information on Mylanta [12] states "Antacids interact with a variety of prescription drugs when taken at the same time. An interaction is unlikely, however, if you keep doses of the two at least (2 or) 3 hours apart."
>
> ***Example 9:*** Information on Lipitor [12] says "If you take Lipitor with certain other drugs, the effects of either could be increased, decreased, or altered. It is especially important to check with your doctor before combining with any of the following: antacids, …" Also included is the statement: "For even greater cholesterol-lowering effect, your doctor may prescribe Lipitor along with different kind of lipid-lowering drug such as Questran or Colestid. It is important to avoid taking two drugs at the same time of the day. Take Lipitor at least 1 hour before or 4 hours after other drugs."

Generalizing from the examples, we see that interaction of a medication with other medications taken by the user may lead to changes in dose size and additional of separation and sequencing constraints. The GMS model captures this information using interaction pairs and their attributes. Depending on the algorithms it uses, a dispenser may combine interaction pairs into subsets of

medications that have similar interactions. The model elements described here for each pair allow the dispenser to derive for each subset the overall constraints and limits of the subset.

In essence, this part of the GMS model can be thought of as an entity-relationship model [13]. Each medication is an entity. Each interaction pair is a relationship between them, and the relationship is specified by the attributes of the pair.

3.1 Attributes of Interaction Pair

Specifically, there is an *interaction pair* $I(M, M')$ for each pair of medications M and M' that may interact to a degree as to require modification in how the medications are to be administered. The *attributes of the interaction pair* contains three components: minimum and maximum separations, direction change lists and precedence constraints.

In the definitions of these components, we sometimes refer to individual dispensing jobs of a medication M as $M(j)$ for j = 1, 2, ⋯ for every j; $M(j+1)$ completes after $M(j)$. It is sometimes more convenient for us to parameterize the dispensing jobs with two indices: We may refer to a job of a medication M as $M(\delta, k)$, where the first index δ refers to the day since the start of the medication. The second parameter k refers to the k-th dispensing job for the medication in day δ.

Minimum and Maximum Separations Between Medications The separation attribute of $I(M, M')$ includes the *minimum separation* $\sigma_{min}(M, M')$ and *maximum separation* $\sigma_{max}(M, M')$ between the medications. To define the terms precisely, we let $M(k)$ be an arbitrary dispensing job of M and t be its completion time. Let $M'(j)$ be the latest job among all dispensing jobs of M' that complete before t, and let $M'(l)$ be the earliest job among all dispensing jobs of M' that complete after t. Figure 4 illustrates these jobs.

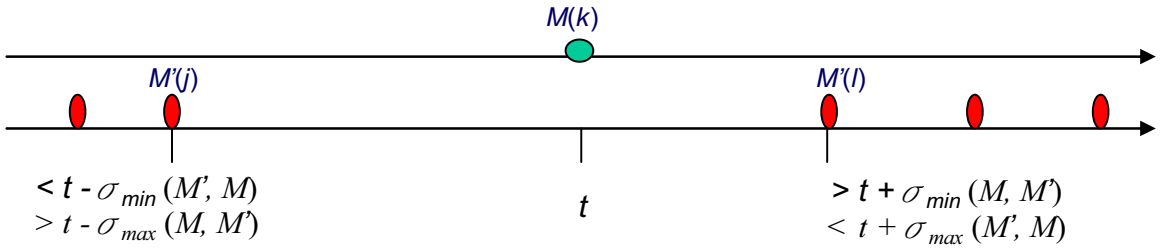

Figure 4 Separations between Doses of Different Medications

If the job $M'(l)$ exists (i.e., whenever M is taken before M'), then the separation from $M(k)$ to $M'(l)$ must be at least $\sigma_{min}(M, M')$. Similarly, if $M'(j)$ exists (i.e., whenever M' is taken before M), $M'(j)$ must be separated from $M(k)$ by at least $\sigma_{min}(M', M)$. The required separation $\sigma_{min}(M, M')$ may not equal to the required separation $\sigma_{min}(M', M)$.

Some medications may be constrained to be taken sufficiently close together or taken together with food. This type of constraint can be expressed in terms of maximum separation between the medications and food. *Maximum separation* can be defined similarly: If the job $M'(l)$ exists, then the separation from $M(k)$ to $M'(l)$ must be no greater than $\sigma_{max}(M, M')$. If $M'(j)$ exists, then the separation from $M'(j)$ to $M(k)$ must be no greater than $\sigma_{max} M', M)$. $\sigma_{max}(M, M')$ may not equal to $\sigma_{max}(M', M)$.

Earlier in Section 2, we mentioned that it is convenient to think of food as a special kind of medication. The constraint that a medication M should be taken on an empty stomach can be specified by separation attributes of the interaction pair $I(M, food)$: The direction in Example 5 states that the minimum separations $\sigma_{min}(Penicillin, food)$ and $\sigma_{min}(food, Penicillin)$ are one hour and 2 hours, respectively. In general, the minimum separations from food are medication dependent. As an example, the direction of Fosamax says that $\sigma_{min}(Fosamax, food)$ is 0.5 hour, not one hour. The constraint that a medication M should be taken together with food can be stated as $\sigma_{max}(M, food) = \sigma_{max}(food, M) = 0$ or some small number such as 0.5 hour.

Returning to Example 7, we observe that for the interaction pair $I(Fosamax, Aspirin)$, the separation from *Fosamax* to *Aspirin* is 1/2 hour or longer. *Fosamax* should be taken first thing in the morning, before the first food, beverage, or other medication [12]. We can account for this requirement by making the minimum separation from all other medications and food to Fosamax so large that it is impossible to schedule a dose of Fosamax on the same day after food and other medications: (Here, we let the minimum separation from everything to *Fosamax* be around 12 hours.) From Example 9, we see that $\sigma_{min}(Lipitor, Mylanta)$ is 1 hour, but $\sigma_{min}(Mylanta, Lipitor)$ is 4 hours.

Examples 7-9 illustrate that the values of minimum separation for an interaction pair M and M' obtained from directions for individual medications may be different. A safe choice for the dispenser is to use the larger of the values. As an example, according to the direction for Mylanta, the minimum separation $\sigma_{min}(Mylanta, Lipitor)$ is 2 or 3 hours, but according to the direction for Lipitor, $\sigma_{min}(Mylanta, Lipitor)$ is 4 hours. It is safer to use 4 hours as the required separation.

Separation Graph The separation relations amongst the medications of the user can be represented by a directed graph. We call such a graph a *separation* (*relation*) *graph*. In the graph, there is a node for each medication M that interacts with some other medication M' taken by the user. There is an edge (M, M') from M to M' if every dose of M must be separated in time from every dose of M' that is taken later. The edge is labeled by $[\sigma_{min}(M, M'), \sigma_{max}(M, M')]$.

To illustrate, Figure 5 shows a separation graph that represent the relations among the

medications of the user in our example. Often, there is no maximum separation requirement (i.e., $\sigma_{max}(M, M')$ and $\sigma_{max}(M', M)$ equal infinity for every interaction pair $I(M,M')$, as is the case in our example. We simply label the edges by the corresponding minimum separations. (In the figure, we also omitted the inequality sign in all but three labels.)

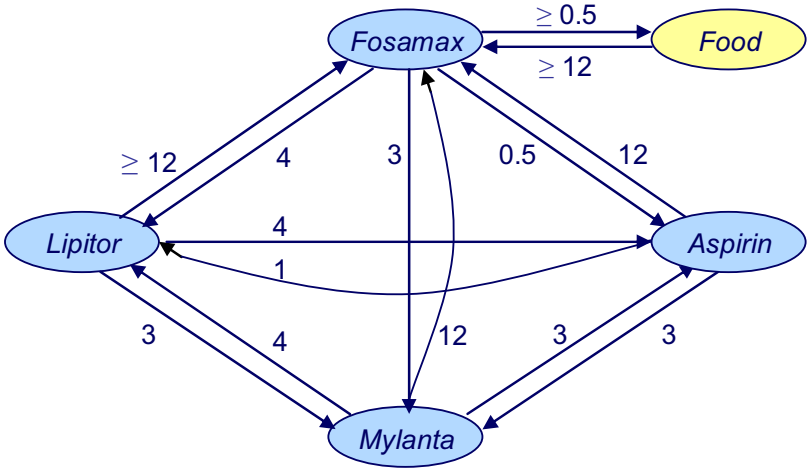

Figure 5 Example of Separation (Relation) Graph (Unit of Time = hr)

Direction Change Lists Some direction parameters for individual medications in an interaction pair may also need to be modified. As an example, if Fosamax is taken alone by a user, the direction may read

Fosamax (10 mg, [∞, ∞], [1, 7], [20, 24], (7, 168), (7, 168)-*Uniform,* TRUE)

where the unit of time is hour. This direction gives the user (and dispenser) the freedom of dividing the 70 mg weekly dose into smaller doses: The user can take 10 mg per day, or 70 mg once a week. Now suppose that the user also needs to take aspirin and is concerned with stomach upset. To minimize the chance of stomach upset, the direction allows only the 10 mg dose size. We can get this constraint by either changing nominal dose size $[d_{min}, d_{max}]$ from [1, 7] to [1, 1] or by changing the maximum intake limit from (7, 168) to (1, 24). In general, the dosages of the medications in each interaction pair may be increased or decreased in order to compensate for the effect of their interactions.

Rather than modifying the direction parameters of individual medications, the model captures the changes in change lists $C(M)$ and $C(M')$ of each of the medications in each interaction pair $I(M, M')$. In our example $I(Aspirin, Fosamax)$, the change list $C(Aspirin)$ for aspirin is empty since no change in the direction for aspirin is required. The change list $C(Fosamax)$ contains either $[d_{min}, d_{max}] = [1, 1]$ or $(B, R) = (1, 24)$. For as long as the user is taking both medications,

the direction parameters given by the change lists replace the parameters specifying the directions for the individual medications.

Precedence Constraint Drug and food interaction of medications may also lead to precedence constraints that dictate the order of their administrations. A dispending job $M(j)$ is a *predecessor* of another job $M'(k)$, and $M'(k)$ is a *successor* of $M(j)$, if $M(j)$ must complete before $M'(k)$. We said that $M(j)$ is an *immediate predecessor* of $M'(k)$, and $M'(k)$ is an *immediate successor* of $M(j)$, if $M(j)$ is a predecessor of $M'(k)$ and there is no other job that is a successor of $M(j)$ and a predecessor of $M'(k)$. We say that the medications M and M' are *precedence constrained* when some jobs for M are predecessors or successors of jobs of M'.

Earlier, we take into account the direction that Fosamax should be taken before any other medication and food every day as separation constraints. A more natural way to take into account this constraint is to make every dispensing job for Fosamax a predecessor of jobs that dispense other medications on the same day. In other words, for every day δ in the duration of medication M and Fosamax, the job *Fosomax* $(\delta, 1)$ that dispense Fosamax on day δ is a predecessor of the job $M(\delta, k)$ that dispenses the k-th dose of M for every k .

Precedence constrained medications may also be separation constrained. Returning to Figure 4, we see that if M is predecessor of M', then $M(k)$ must be an immediate predecessor of $M'(l)$ in the subset of precedence relations of dispensing jobs for M and M'. (In other words, $M(k)$ is a predecessor of $M'(l)$ and there are no other jobs for M and M' that are successors of $M(k)$ and predecessors of $M'(l)$.)

3.2 Interaction (Non-Compliance) Event

We call a violation of one or more constraints specified by some attributes of an interaction pair an *interaction* (non-compliance) *event* or an event of *interaction event type*. Because the GMS specification treats food as a special medication, an interaction event may occur even when the user is taking only one medication.

Many causes can lead to an interaction event: the violated constraint can be specified by a change list, or the separation graph, or a precedence relation. We divide the interaction event type into subtypes accordingly. Take the user who is on both Fosamax and aspirin as an illustrative example. The user may choose to take 70 mg of the former once a week, trading off convenience at the risk of upset stomach. By so doing, the user violates the direction given by the change list of Fosamax: The list specifies either the reduced dose size [1, 1] or the smaller maximum in-take (1, 24). Therefore this interaction event is also an over-dose (or over-medicate) event. In addition

to *interaction/over-dose* and *interaction/over-medicate*, the subtypes arising from violations of constraints specified by change lists also include *interaction/under-dose* and *under-medicate*.

An interaction event may also be a violation of one or more constraints defined by some edges in the separation graph. It is of the *interaction/concurrency* subtype. Similarly, when some medications are constrained to be taken in some order (e.g., Fosamax before everything else), an interaction event violating a precedence constraint is of the *interaction/sequencing* subtype.

4 User Preference Parameters

We concentrate here on the user preference and life style parameters that are part of the GMS model and specification. In general, these parameters also provide information that is unrelated to medication scheduling. A smart dispenser can customize for the user other aspects of its operations and behavior based on the information. As an example, a direction of Linoxin says that the doctor should be consulted when an under-medicate event occurs. So, the action "send an alarm" is a direction parameter. A user may want the alarm sent to himself/herself, a caretaker or directly to a physician in this situation. This choice is a preference parameter and is specified by the user. We defer discussions on aspects of customization that are unrelated to medication scheduling to a later report.

4.1 Calendar Interface

From usability perspective, a smart dispenser must aim to make the medication schedule it generates for the user flexible and easy to follow. This aspect of usability is especially important for elderly users. A seriously ill user on life-critical medications for a relatively short time may be willing to synchronize sleeping and eating hours with a rigid medication schedule. A user who is taking medications for months and years is unlikely to be willing and able to do so, especially when the user is active and busy every day. A smart dispenser for such a user must try to fit the medication schedule to user's daily and holiday routines whenever possible. When some adjustment in daily routines must be made for compliance's sake, a smart dispenser should be able to explain to the user why such changes are necessary and work with user to make the least objectionable adjustment. A rigid, hard-to-live-with dispenser risks being discarded at worst, being often ignored at best.

The user can provide several user preference, lifestyle, and behavior parameters to help the dispenser function better in this respect. The specifics of how a smart dispenser collects and updates these parameters are beyond the scope of this report. It suffices for us to assume here that the dispenser provides some kind of calendar function for this purpose. In the simplest form,

the dispenser calendar interface may resemble the ones commonly provided by household thermostats[3]: Following a simple menu, the user can enter via a keypad typical wake and sleep times for every day of the week at initialization time and when significant changes in daily routines occur. The dispenser can also collect in similar way information on typical number and times of meals and snacks, in weekdays, weekend and holidays. The user should also be able to provide most preferred intervals of time for medication as well as times when reminders and medications are likely to be unwelcome.

Many user behavior parameters affect scheduling and compliance monitoring. An important one is *promptness*. (The term was defined in Section 1 as the length of time the user takes to come and retrieve a medication after being reminded it is time for a dose.) A dispenser must take into account the user's promptness in scheduling the start of each dispensing job. Promptness varies. The user can provide only rough estimates of its value and range. The dispenser must automatically collect statistics on this parameter while it is in use. So, over time and through use, the dispenser will be able to better estimate user's promptness.

A more sophisticated dispenser may also have some mechanism to collect statistics on the user's actual wake, sleep, meal and snack times. Statistics on these user lifestyle parameters can help the dispenser to better estimate the actual times when the user is willing and can take medications. Indeed, we envision that a future household is likely to deploy multiple SISARL (Sensor Information Systems of Active Retirees and Assisted Living) devices [14], in addition to smart household appliances. The devices can collaborate in collecting such information since the information can help many of them to function better.

4.2 Feasible and Forbidden Intervals

Rather than keeping track of ranges of wake, sleep and meal times of the user, the GMS model represents this information in a form most convenient for the scheduler. After capturing these times, the dispenser scheduler uses them, together with direction parameters related to user's daily routines, to divide the time line into feasible intervals and forbidden intervals.

Definitions and Illustrating Examples A time interval (*t, t'*) is *a feasible interval of a medication* if the dispenser is allowed to start dispensing jobs of the medication in the interval.

---

[3] Users in areas where houses are heated in winters or cooled in summers are likely to be familiar with thermostat interface. In addition to wake and sleep times, the user also provides the temperature settings during sleep and wake hours. Thermostat interfaces typically allow the user to enter a specific time of each type (e.g., wake time is given as 6:10 AM.) Dispensers may allow the user to give a range of time (e.g., wake time given as (6:00 AM – 6:20 AM)) or simply assume that by a specific time, the user means by default that time plus-minus 10 minutes.

Time intervals in which no dispensing job of the medication is allowed to start are *forbidden intervals*.

To illustrate, Figure 6 shows the intervals of typical sleep, meals and snack times that a user may have entered when prompted by the calendar interface of the dispenser. By default, the user does not want to take anything during sleeping hours from late evening to early morning. Suppose that the user is taking nothing but vitamin and calcium supplements. Their directions say to take one tablet once or twice daily with food. For these supplements, there is only one forbidden interval, and that is the user's sleeping hours.

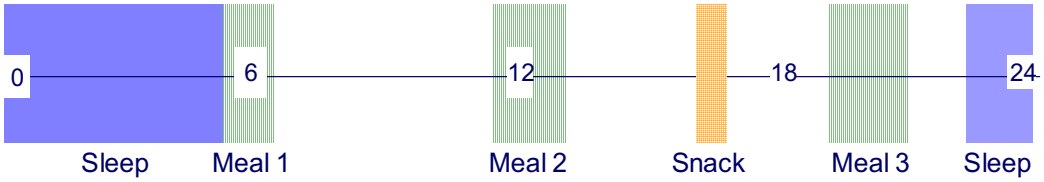

Figure 6 Sleep, Meal and Snack Hours

The intervals marked as meal and snack times are feasible intervals. A dispenser may prioritize the available feasible intervals: It schedules a dispensing job in a lower priority feasible interval only if some compliance constraint would be violated if the job were scheduled in higher priority intervals. Depending on the algorithms it uses for scheduling, the dispenser may assign priorities to feasible intervals in many ways. It may use user convenience and medical merits as basis. For example, the dispenser in Figure 6 may give the morning and evening meal times the highest priority and the intervals during noon-time meal and snack a lower priority because the user is more likely to be away from home and the dispenser during the middle of the day. The dispenser may also use priority as a means to tune the performance of the scheduler; the dispenser may prioritize the intervals so as to increase some figures of merit of the resultant schedule. As examples, the dispenser may give earlier intervals higher priority, or long intervals higher priority, or shorter intervals higher priority, etc. in order minimize the rate of failure to meet constraints.

While it is desirable to take vitamin and calcium with food, it is safe to take them at other times. So, they can also be dispensed in time intervals between meals and snack if necessary. The dispenser may mark these intervals as feasible. We call them *backup* (*feasible*) *intervals*; they have the backup (lowest) priority.

In general, a medication may not have any backup feasible interval. To illustrate, we suppose that the user in the previous example is not well and needs to take a medication, and its direction says "take one dose every 5 to 6 hours on an empty stomach (i.e., either one hour before or 2

hours after meals and snacks) for 4 times a day." The requirement that the medication be taken on an empty stomach narrows down feasible intervals to the ones indicated by the dotted boxes on the upper time line in Figure 7. All the remaining times are in forbidden intervals.

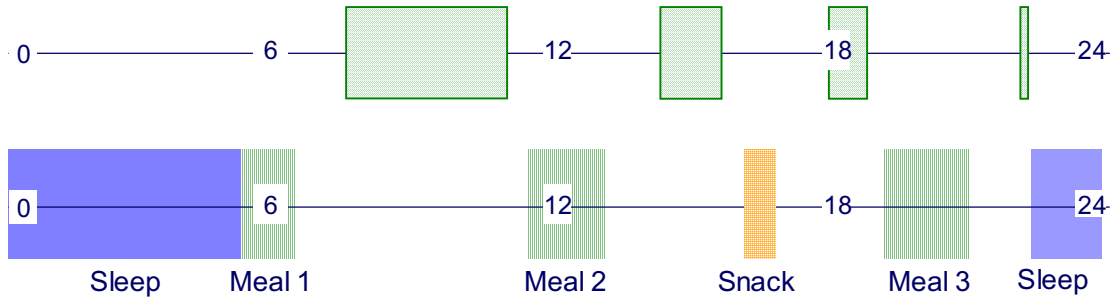

Figure 7 Example Illustrating Need for Adjustment in Daily Routine

In this case, it is not possible to schedule 4 doses 5 to 6 hours apart without some change in user's daily routine. A user who can return to sleep easily may let the dispenser schedule a dose during sleeping hours. On the other hand, a user who needs uninterrupted sleep may prefer to adjust meal and snack times if doing so can create additional feasible intervals or lengthen existing ones to accommodate the required doses. A smart dispenser should compute and present to the user these alternatives and help the user to make the least objectionable choice.

Similarity and Difference with Existing Models Feasible interval is a commonly used term in traditional real-time workload models [5]: According to existing models, the feasible interval of a job begins when the job is ready to be scheduled and ends at the deadline of the job. A well-designed job can meet its deadline surely or with a high probability if it is scheduled to execute alone in a feasible interval. If it were scheduled in a forbidden interval, the job would surely miss its deadline. Here is where resemblance between feasible intervals in the GMS model and existing models ends. Feasible intervals in the GMS model hold no such promise since feasibility is based on user preference, not timing parameters. The dispenser may not be able to schedule is a job in a feasible interval because some constraint of the job would be violated if the job were scheduled there.

Feasible intervals in the GMS model resemble feasible intervals in the state-dependent deadline model [9]. The problem of scheduling dispensing jobs of a medication in multiple feasible intervals of the medication is similar to problems in scheduling state-dependent deadline jobs with multiple feasible intervals [10]. The state-dependent deadline model was motivated by the fact that the deadlines of some real-time monitor and control jobs depend on the states of the objects under their observation and control. When states of some objects are not completely

known and may even change with time, the deadlines of the jobs are also not completely known and may even change with time. A dispenser is indeed such a system: It monitors and "controls" the user. Since the user's daily routines and behavior cannot be predicted completely, the time intervals most convenient for the user to take medications are uncertain.

5 GMS Graph Representation

As a way to summarize the GMS model elements presented in previous sections, we present here a graph representation of medication schedules generated from GMS specifications. We call a graph that gives a partial representation of a GMS schedule a *GMS graph*.

5.1 Elements of GMS Graphs

As in traditional task graphs, nodes in a GMS graph represent jobs. Edges in the graph represent relationship among jobs.

Dispensing Jobs: Specifically, a node represents a dispensing job of a medication *M*. As defined Section 1, each job starts by sending a reminder to the user to take a dose of *M* and ends when the dispenser detects and marks the retrieval of the dose. These instants are start time and completion time of the job. From the dispenser's perspective, a dose of *M* is taken (or dispensed and administered) at the completion of a dispensing job.

The *elapse time e* of a dispensing job is the length of time between the start time and completion time of the job. It constitutes mostly of the promptness of the user. Since the exact value of promptness is not known, we treat the elapse time *e* as a random variable. Its probability density function (PDF) $p_e(0)$ is estimated by an even distribution when the dispenser is first put to use at time $t = 0$. (The even distribution is specified by an average value and a range of value provided the user at initialization time, for example.) By measuring user promptness at each dispensing, the dispenser can get better and better estimate of the PDF $p_e(t)$ at time $t > 0$ while the dispenser is in use.

An assumption of the GMS model is that the dispenser is well designed: It has sufficient processing, storage and communication power and, hence, can send messages, monitor and record user behavior, check for compliance, sound alarm, etc. in a fraction of a second. This time is negligibly small compared with user promptness. It follows that elapse times of dispensing jobs of all medications are identically distributed. They are not statistically independent: A user is more likely to be slow for one job if he/she was slow for earlier jobs, for example.

In a GMS graph, we label each job of a medication by $M(\delta, k)$ when we want to keep track of

the day in which the medication is administered. The *release time* of a job is the time instant when the job is available for scheduling. The release time of $M(\delta, k)$ is the beginning of the first feasible interval in day δ. In case of a medication that is to be taken night and day regardless or does not have the same number of doses each day, we may refer to its dispensing jobs as $M(k)$, for $k = 1, 2, \ldots$ The release times of these jobs are the start of the direction duration.

Task or Medication Following the convention of traditional real-time workload models, we call a sequence of dispensing jobs of a medication a *task* and name the task by the name of the medication. For example, Lanoxin is a sequence of dispensing jobs that administers Lanoxin. When there is no ambiguity, we also call a task a *medication*. So, we may refer to the sequence of jobs dispensing Lanoxin as the task Lanoxin or medication Lanoxin in different contexts.

A medication that has a time varying direction is a task that consists of a sequence of subtasks, a subtask per static direction in the time varying directions. Dispensing jobs in each subtask have the same direction and user preference parameters. In particular, all the jobs have the same dose size parameters. They have the same temporal separation constraints. Together, they must satisfy the same maximum and minimum total intake constraints.

Similarly, all dispensing jobs of a medication have the same feasible intervals and forbidden intervals. This fact follows from the definitions of the intervals.

Relationships of Jobs In the GMS graph, we represent a separation constraint between two jobs $M(j)$ and $M'(l)$ by an undirected edge when the jobs can complete in any order. The edge is labeled by the ranges $[\sigma_{min}(M, M'), \sigma_{max}(M, M')]$ and $[\sigma_{min}(M', M), \sigma_{max}(M', M)]$ when the ranges are finite.

Precedence constraint between two jobs $M(j)$ and $M'(l)$ is represented by a directed edge from $M(j)$ to $M'(l)$ if $M(j)$ is an immediate predecessor of $M'(l)$. The edge is also labeled by the range $[\sigma_{min}(M, M'), \sigma_{max}(M, M')]$ of temporal separation between the jobs. Precedence constraint between two arbitrary dependent jobs is represented by a directed path from the predecessor to the successor.

When a job $M(j)$ is an immediate successor of a $M'(k)$ and their separation must be in the range $[s, s']$, maximum separation s' between their completion times gives the latest time the successor must complete with respect to the completion time of the predecessor. The length of time s' is the *relative deadline* of $M'(k)$. The relative deadline of a job with multiple predecessors is equal to the minimum of its maximum separations with respect to the predecessors.

Figure 8 gives an example. The user takes three medications A, P and S. The precedence

constraint between consecutive jobs in each of the medication follows directly from their definition. The (dotted) undirected edge between the job $S(k)$ and last job of the day for A indicates that the doses of the medications can be taken in any order provided they are separated sufficiently far apart and that the required separation is independent of the order the medications are taken.

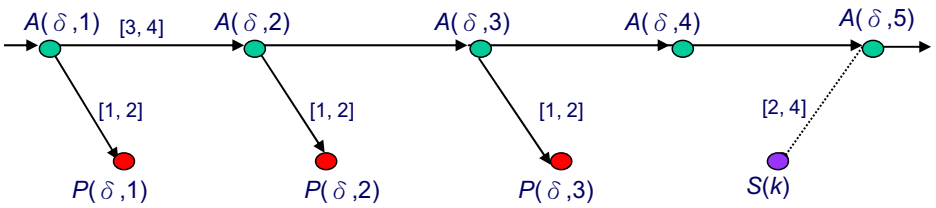

Figure 8 Example of GMS Graph

Resource Models As stated earlier, a smart dispenser has sufficient system resources to carry out dispensing jobs of multiple medications at the same time. Also, the user can respond to multiple reminders and come to retrieve multiple medications. Hence, it is reasonable for the dispenser scheduler to assume a "resource rich" model: That is, any number of dispensing jobs can proceed at the same time if the jobs are not forced to proceed at different times by separation and intake constraints.

Alternatively, the dispenser may treat the user as a resource that may not be available sometimes: Some medications are allowed to share this resource freely, while others require exclusive use. In the previous section, we capture the requirement that a medication be taken on an empty stomach by putting one hour before and two hours after every meal and snack in forbidden intervals. By doing so, we make the user a "plentiful resource" because it can be shared by all dispensing jobs. In the scarce resource view of the user, these intervals of time remain in feasible intervals. However, the medication requires exclusive use of the user for one hour, and food requires exclusive use of the user for two hours. In other words, a dispensing job of the medication cannot start and end within one hour of a meal or snack, and eating cannot start (because the user is not available) within two hours after the completion of a dispensing job. This model is likely to give the dispenser more scheduling flexibility. The scheduling problem in the resource poor model is more complex, however.

5.2 Comparison with Traditional Task Graphs

Despite many similarities between them, GMS graphs are not traditional task graphs [5]. Their differences arise from their intended use and the underlying resource models.

In essence, a task graph is a partial, abstract description of time-critical applications on a given platform in an embedded, real-time system: It is a representation of the workload imposed by the application tasks on the underlying platform. Underlying the representation is a model of available system and application resources, together with rules governing how the resources are to be allocated when applications content for them. The task graph specifies not only the timing constraints of the applications and their components, but also the resource demands of the components. Most existing techniques for validating timing constraints rely on this and similar kinds of workload description.

A medication dispenser is an embedded real-time system. Its workload consists of computation and communication tasks that generate and dynamically adapt the medication schedule, send reminder messages, monitor user behavior, check for compliance, and so on. These tasks need resources such as processor, memory, network link to make progress. Many of the tasks are time-critical. We surely can use of a task graph to capture the timing and resource requirements of the workload of the dispenser.

GMS graph does not represent the work done by the dispenser. Rather, it is a specification of how the dispenser should do its work.

6 Summary

This report describes the GMS model for specifications of medication schedules. A key assumption here is that someone (e.g., physician and/or pharmacist) has already verified that it is safe for the user to take all the medications managed by the dispenser, provided that all the medications are taken as directed. The direction parameters are correct in the sense that ill effects of drug interactions are known and tolerable. The primary goal of the dispenser is to help the user following the directions.

A part of the work by a dispenser is to compute a schedule of its own commands and dynamically adapting the schedule as needed in response to user actions. By following the schedule and carrying out its commands, the dispenser works with the user in the administration of the medications under its control in order to prevent non-compliance. A GMS specification defines rigorously for the dispenser the constraints in scheduling and adaptation. Non-compliance events of more or less serious natures may nevertheless occur. The dispenser must be able to detect, categorize and record these events and take actions that are appropriate for the medication, the seriousness of non-compliance, and the user. The information it needs to do this part of the work is also captured by the GMS specification.

Many intended users of the dispensers are elderly individuals. Some of them still live active and busy lifestyles despite being on health-critical medications. A dispenser may be in use for many years. It is not reasonable to expect a user to follow a rigid schedule faithfully for long periods of time. Therefore, the dispenser must provide the user with sufficient flexibility in compliance, allowing the user to deviate from correct schedules now and then. How to do so in a provably safe way is a challenge, and overcoming this challenge is an objective of our work.

Much of work on the development of the GMS model remains to be done. We are current developing algorithms for consistency check. The parameters that guide normal operations of the dispenser and define compliance are not independent. An example involves the required schedule duration T of a medication and parameters defining the minimum intake constraint (L, P): For every medication, consistency between the two constraints demands that T be zero if and only if the minimum intake is zero (i.e., (0, P)) and that T be no less than P in general. A consistency check algorithm can detect any inconsistency between constraints defined by the given parameters. When inconsistencies are detected, the algorithm can correctly modify the parameters in order to remove the inconsistencies. Our algorithms are extensions of existing algorithms for checking consistency among timing constraints of the real-time tasks.

We also need to clearly understand how nearly complete the model is. By nearly complete, we mean that the model can capture requirements and constraints defined by directions of essentially all medications. This work is a part of our effort on an authoring tool. One of its functions is to extract directions from user's prescriptions and on-line drug libraries and generate a GMS specification for use by medication dispensers. The architecture and design of the tool and its implementation will be detailed in a future report. In the process of evaluating the tool, we will also evaluate the GMS model for completeness.

References

[1] S. C. Dursco, "Technological Advances in Improving Medication Adherence in the Elderly," *Annals of Long-Term Care: Clinical Care and Aging*, Vol. 9, No. 4, 2001.

[2] Pill boxes and medication scheduling at http://www.epill.com/ and http://www.dynamic-living.com/automated_medication_dispenser.htm

[3] My Pill Box at http://www.mypillbox.org/mypillbox.php

[4] G. Gerguson, J. Allen, N. Blaylock, D. Byron, N. Chambers, M. Dzikovska, L. Galescu, X. Shen, R. Swier, and M. Shift, "The Medication Advisor Project: Preliminary Report," Report No. 776, Computer Science Department, University of Rochester, May 2002.

[5] J. W. S. Liu, *Real-Time Systems*, Chapters 2 and 3, Prentice Hall, 2000.

[6] B. Spruri, L. Sha, J. P. Lehoczky, "Aperiodic task Scheduling for Hard Real-Time Systems," *Real-Time Systems Journal*, Vol. 1, No. 1, 1989.

[7] J. W. S. Liu, K. J. Lin, W. K. Shih, R. Bettati and J.Y. Chung, "Imprecise Computations," *IEEE Proceedings*, Vol. 82, pp. 1-12, January 1994.

[8] M. Hamdaoui and P. Ramanathan, "A Dynamic Priority Assignment Technique for Streams with (m, k)-firm deadlines," *IEEE Transactions on Computers*, Vol. 44, No. 12, December 1995.

[9] C. S. Shih and J. W. S. Liu, "Scheduling State-Dependent Jobs," *Proceedings of 2002 IEEE Symposium on Real-Time Systems*, pp. 3-14, December 2002.

[10] C. S. Shih, J. W. S. Liu and I. Cheong, "Scheduling Jobs with Multiple Feasible Intervals," *Proceedings of 2003 RTCSA,* pp. 213-231, February 2003.

[11] P. H. Tsai, P. C. Hsiu, H. C. Yeh, , C. S. Shih, J. W. S. Liu, T. W. Kuo, and T. Y. Huang, Report on dispenser scheduling and non-compliance event handling in preparation.

[12] PDRHealth, Drug Information, http://www.pdrhealth.com/drug_info/.

[13] Proceedings of Int. Conference on Conceptual Modeling, Formally Int. Conference on ER Approach, http://www.informatik.uni-trier.de/~ley/db/conf/er/

[14] J. W. S. Liu, *et al.*, "Reference Architecture of Intelligent Appliances for the Elderly," *Proceedings of the 18th International Conference on System Engineering,* August 2005.

Appendix Partial List of Terms and Notations

The appendix provides definitions and notations of the terms used in the report.

Absolute minimum separation S_{min} and absolute maximum separation S_{max}: Values delimiting the compliant range of temporal distance between consecutive doses.

Absolute minimum dose D_{min} and absolute maximum dose D_{max}: Values delimiting the range of compliant dosage.

Change list $C(M)$: A list containing changes in direction of medication M necessitated by interaction of M with other mediations and food.

Completion time of a job: The time instant when the user retrieves a dose of the medication that is dispensed by the job.

Concurrency event: A violation of the instruction that two or more actions must be taken (or must not be taken) at the same time.

Direction parameters: GMS parameters that are derived from directions of medications.

Dispensing job (or **job**) **for a medication:** A sequence of actions by the dispenser to administer a dose of the medication:

Duration: The length of time or total number of doses (delimited by the minimum duration T_{min} and the maximum duration T_{max}) over which direction parameters of a medication are to be followed.

Elapse time of a job: The length of time between the start time and completion time of the job.

Feasible interval: A preferred interval time for dispensing a medication.

Forbidden interval: A time interval in which no dispensing job is allowed to start.

GMS graph: A graph that represents a GMS schedule.

GMS specification: A specification of medication schedules based on the GMS model described in the report.

Granularity: The size of unit in term which dosage of a medication is specified.

Interaction event: A violation of a constraint arisen from interaction of a medication with other medications and food.

Interaction pair *I* (*M, M'*): A part of the GMS specification that defines changes in directions of medication *M* and *M'* due to interactions between the medications.

Maximum total intake (*B, R*): A limit (defined by a budget *B* and replenishment time *R*) to total dosage *B* over any time interval of length *R.*

Minimum total intake (*L, P*)-Periodic: Time being segmented into periods of length *P,* a limit *L* to the minimum total dosage taken in every period.

Minimum total intake (*L, P*)-Uniform: A limit *L* to the minimum total dosage taken within any time interval of length *P.*

Nominal minimum dose size d_{min} and maximum dose size d_{max}: Dose sizes delimiting the

range of dosage dispensed each time.

Nominal minimum separation s_{min} and maximum separation s_{max}: Values delimiting the range of temporal distance between consecutive dose.

Over-dose event: An event of the user taking a dose of size larger than the absolute maximum dose size, or two doses separated in time less than the absolute minimum separation, or both.

Precedence constraint: A constraint in the order of dispensing jobs.

Predecessor (or successor): A dispensing job that must complete before (or after) another job.

Over-medicate event: A violation of the maximum intake constraint or the maximum duration constraint.

Promptness: The amount of time the user takes to come and retrieve a medication after being reminded to take a dose.

Smart (medication) dispenser: A programmable device designed to help its user to compliant to medication directions and to provide the user or caretaker with warnings when it detects non-compliance.

Start time of job: The time instant when the dispenser reminds the user to come a take a dose of medication.

Static direction: A medication direction defined by constant parameters.

Time instant when a medication is administered or dispensed: The completion time of an associated dispensing job.

Time varying direction: A sequenced set of static medication directions.

Under-dose event: An event of the user taking a dose of size less than the absolute minimum dose size $D_{min,}$ or two consecutive doses further apart than the absolute maximum separation. S_{max}, or both.

Under-medicate event: A violation of the minimum intake constraint or the minimum duration constraint.