J Med Syst (2012) 36:2993–3009
DOI 10.1007/s10916-011-9777-z

ORIGINAL PAPER

Triggers and Monitoring in Intelligent Personal Health Record

Gang Luo

Received: 11 July 2011 / Accepted: 22 August 2011 / Published online: 23 September 2011

Abstract Although Web-based personal health records (PHRs) have been widely deployed, the existing ones have limited intelligence. Previously, we introduced expert system technology and Web search technology into the PHR domain and proposed the concept of an intelligent PHR (iPHR). iPHR provides personalized healthcare information to facilitate users' daily activities of living. The current iPHR is passive and follows the pull model of information distribution. This paper introduces triggers and monitoring into iPHR to make iPHR become active. Our idea is to let medical professionals pre-compile triggers and store them in iPHR's knowledge base. Each trigger corresponds to an abnormal event that may have potential medical impact. iPHR keeps collecting, processing, and analyzing the user's medical data from various sources such as wearable sensors. Whenever an abnormal event is detected from the user's medical data, the corresponding trigger fires and the related personalized healthcare information is pushed to the user using natural language generation technology, expert system technology, and Web search technology.

Keywords Personal health record · Trigger · Monitoring · Natural language generation · Search engine · Expert system

G. Luo (✉)
IBM T.J. Watson Research Center,
19 Skyline Drive,
Hawthorne, NY 10532, USA
e-mail: luog@us.ibm.com

Introduction

Web-based personal health records (PHRs) enable consumers to actively manage their medical records and subsequently their health through a Web interface. PHRs are becoming increasingly important in the healthcare business, as evidenced by their wide deployment by several major Internet companies, including Microsoft [32], WebMD [31], and RelayHealth [40]. Nevertheless, the existing PHRs have limited intelligence.

To improve PHR's capability of meeting users' healthcare needs, we previously proposed the concept of an intelligent PHR (iPHR) [29]. The main idea is to introduce expert system technology and Web search technology into the PHR domain so that iPHR can provide personalized healthcare information to facilitate users' daily activities of living. In addition to a standard PHR system, a medical knowledge base, an expert system, and a search system are also essential components of iPHR. Average consumers typically have little medical knowledge and encounter difficulty in identifying appropriate health-related keywords. iPHR addresses this problem by extensively using medical knowledge to both guide users to provide the most important information about their medical condition and automatically form queries. These queries are termed "search guide information" and reflect the user's medical condition and healthcare needs. They serve as seeds for the search system to retrieve personalized healthcare information.

Our current iPHR system provides three functions: guided search for disease information [26], recommendation of home nursing activities [28], and recommendation of home medical products [30]. It follows the pull model of information distribution. When a user wants healthcare information, she logs into iPHR and invokes a function that

can provide this information. As mentioned in [29], this passive style is sub-optimal. To better utilize its potential, iPHR should be able to automatically detect when the user needs relevant, personalized healthcare information and then actively pushes this information to her.

In this paper, we introduce triggers and monitoring into iPHR so that iPHR can become active. Our key idea is that through monitoring users' medical data and detecting abnormal events that may have potential medical impact, iPHR identifies when users need to be aware of relevant, personalized healthcare information. iPHR can then push healthcare information related to the abnormal events to users accordingly.

More specifically, medical professionals pre-compile triggers and store them in iPHR's knowledge base. Each trigger corresponds to an abnormal event. iPHR keeps collecting, processing, and analyzing the user's medical data from various sources such as wearable sensors [15]. Whenever an abnormal event is detected from the user's medical data, the corresponding trigger fires and the related personalized healthcare information is pushed to the user in its basic form using medical knowledge and natural language generation technology [41]. If the user clicks the hyperlinks embedded in the basic healthcare information, iPHR will first use medical knowledge and expert system technology to automatically form the corresponding queries, then use these queries and Web search technology to retrieve more detailed healthcare information, and finally present this retrieved information to the user. The entire process of monitoring users' medical data, detecting abnormal events, and pushing related personalized healthcare information to users is tightly integrated, automated, and easily usable by average consumers with little medical knowledge.

The design principle of iPHR with triggers is different from that of traditional electronic health records. Traditional electronic health records focus on data storage and primarily use data storage, data retrieval, data communication, and data visualization technologies. In comparison, iPHR with triggers has an additional focus on data processing and uses additional technologies involving medical signal processing [45], knowledge-based reasoning (to integrate the results of signal analysis along with other discrete data), natural language generation, expert system, and Web search. As a result, iPHR with triggers can provide more advanced functions and meet more healthcare needs of consumers than traditional electronic health records.

The rest of the paper is organized as follows. "Motivation for triggers and monitoring in iPHR" provides the motivation for triggers and monitoring in iPHR. "Design of triggers and monitoring in iPHR" describes the design of triggers and monitoring in iPHR. "Experimental results" presents a few output examples of our active iPHR system. "Related work and discussions" discusses related work. "Conclusions" concludes this paper.

Motivation for triggers and monitoring in iPHR

In this section, we first provide the general motivation for introducing triggers and monitoring into iPHR, and then present two concrete scenarios illustrating what this new trigger function of iPHR can do.

General motivation

Today's society is facing a healthcare crisis due to demographic aging and a global shortage of healthcare workers [50]. By 2050, the population over 60 is expected to triple. At the same time, the potential support ratio, which is defined as the size of the population aged 15–64 divided by the size of the population over age 65, is expected to drop by half and reach the dangerously low level of 4:1 [47]. To make the situation worse, chronic conditions have already become widespread while requiring expensive, long-term medical attention. Ninety percent of seniors have at least one chronic condition. Twenty-one percent of Americans and 77% of seniors have two or more chronic conditions [3, 48]. To make our healthcare system sustainable, we must invent new ways to perform healthcare, e.g., by empowering consumers to take an active role in managing their health and by putting more emphasis on prevention and health maintenance.

In particular, in the community setting, we need to help patients increase their independence in managing chronic diseases as much as possible so that the amount of attention required from healthcare professionals can be dramatically reduced. An essential step to this end is to keep monitoring and assessing the patient in the community setting with little or no involvement of healthcare professionals. The goal is to detect potential health problems early on and then perform interventions in time to reduce the likelihood and/or the speed at which the patient's condition deteriorates. In general, a degraded condition will both increase the overall healthcare cost and reduce the patient's quality of life. It would be better to notify the patient about her potential health problems and provide immediate feedback about what can be done for them at an early stage.

As mentioned in [38], most of the monitoring, feedback providing, and education of patients with chronic conditions involves a limited number of educational goals and physiological measurements. It can be accomplished via automatic tools in the absence of healthcare professionals. This paper introduces triggers and monitoring into iPHR to serve this purpose.

For effective patient monitoring, it is not sufficient to rely solely on patients' subjective reporting. Patients do not have an objective perception of their own status, such as the severity of their symptoms or their response to medication, and thus cannot provide reliable reporting. In particular, physiological changes frequently precede a clinical deterioration, but could go unnoticed by the patient. For example, asthma patients may have an impaired sensation of shortness of breath [49]. As another example, patients with heart failure may subconsciously adapt their activity level to compensate for their decreased activity tolerance [38]. In both cases, patients can be unaware of their deteriorating health because their physiological responses effectively mask their worsening symptoms.

To detect early changes related to clinical deterioration, we need to monitor the patient by performing ongoing, objective, physiological evaluation of her, regardless of whether she has complained. This can be done using wearable sensors [15] and various other personal medical monitoring devices, such as scale and peak flow meter. All medical data collected on the patient can be sent to and stored in her PHR in a relatively streamlined manner. Recall that a standard PHR system is included in iPHR. Nevertheless, it is nontrivial to make effective use of this data.

In practice, the amount of collected medical data can quickly become substantial. For instance, several days of continuous, physiological time series signals, such as electrocardiogram and electroencephalogram, are often multiple megabytes to multiple gigabytes in size. Nevertheless, the detection of many potential health problems requires enough evidence to accumulate over an extended period of time (e.g., a month). If presented with all of the data collected over that time period without any pre-processing, even with the help of the best data visualization techniques, healthcare professionals will easily be overwhelmed and unable to make appropriate decisions in a reasonable amount of time [42]. If we consider average consumers, the outlook is much worse because they typically have little medical knowledge and cannot interpret the collected data at all.

To make good use of collected medical data, we often need to pre-process it to filter out irrelevant details and extract the essential information. This information is used to identify when users have potential health problems. In this work, each potential health problem that can be recognized by iPHR is described as one or more abnormal events. Whenever an abnormal event is detected from the medical data collected on the user, iPHR anticipates that she will need healthcare information related to this abnormal event and automatically pushes that information to her. In this way, the user gets notified of both her potential health problem and how it can be managed, ideally at an early stage.

Triggers

Embedding triggers into iPHR is a general approach for automatically detecting abnormal events. Each trigger corresponds to a unique abnormal event $E_{abnormal}$ and is of the form

$$\text{Firing condition } C_f \rightarrow \text{Action } A \text{ (triggering event } E_{triggering}, \text{ applicable condition } C_a),$$

meaning that if the applicable condition C_a applies to the user, then the action A will be taken when the triggering event $E_{triggering}$ occurs and the firing condition C_f describing $E_{abnormal}$ is satisfied.

Typically, the triggering event is that a new piece of data used in the firing condition C_f is collected on the user. In this case, C_f determines the triggering event. Other kinds of triggering events, such as the elapse of time (e.g., once every year), are also possible.

Examples of an applicable condition include when the user has a particular disease, the user is taking a particular medicine, the user is above a certain age, and the empty condition that applies to each user.

The simplest action, which is also the most common one, is to report the abnormal event $E_{abnormal}$ to the natural language generation system of iPHR. Then iPHR can push healthcare information related to $E_{abnormal}$ to the user accordingly. A more complex action is to report different levels of $E_{abnormal}$ based on the range that a computed value falls into [21]. Another more complex action is to use a rule-based algorithm to ask additional questions if $E_{abnormal}$ is suspected whereas extra inputs are needed to confirm this suspicion [38], and then make a final decision on whether $E_{abnormal}$ should be reported.

After data pre-processing, a trigger can synthesize the important information from one or more data sources to report a particular abnormal event. The firing condition of the trigger can be either simple or arbitrarily complex. For many complex firing conditions, such as the ones involving nontrivial quantitative computation, checking them is beyond the manual capability of both healthcare professionals and average consumers in a reasonable amount of time, but usually can be done by computers (almost) in real time.

In the following, we provide two concrete scenarios illustrating what the new trigger function of iPHR can do.

Scenario 1: Weight loss in COPD patients

Chronic obstructive pulmonary disease (COPD) affects about 24 million Americans and is a leading cause of morbidity and mortality worldwide [10]. It is characterized by airflow limitation that is usually progressive and not fully reversible.

COPD patients often experience weight loss, which is associated with increased risks of mortality, disability, and handicap. Criteria to define weight loss is losing >5% weight in the past month or losing >10% weight in the past six months [9]. It would be desirable to keep track of the weight of a COPD patient, e.g., by using a scale and recording his weight in his PHR on a periodic basis. When weight loss is detected, it would be beneficial for iPHR to provide him with the following basic healthcare information:

(1) COPD patients experiencing weight loss may need nutritional therapy.
(2) Since weight loss in COPD patients is often accompanied by muscle wasting, nutritional therapy may only be effective if it is combined with anabolic stimuli such as exercise.

It is also possible that he is interested in knowing more details about one or more specific items in the basic information, such as nutritional therapy for COPD patients and exercises for COPD patients.

In this scenario, the weight reading of the COPD patient can be recorded in his PHR in the format of (weight w_i, date d_i), meaning that his weight was w_i on date d_i. Let (W, D) represent his current weight reading, i.e., his weight is W on the current date D. Then the maximum weight recorded on him in the last n days is

$$max\_w_n = \max_{d_i \geq D-n} w_i.$$

In particular, the maximum weight recorded in the past month is max_w_{30}. The maximum weight recorded in the past six months is max_w_{182}.

For the abnormal event of weight loss in the COPD patient, we define a corresponding trigger whose firing condition reflects the defining criteria of weight loss and therefore is

$$(W < 0.95 \times max\_w_{30}) \vee (W < 0.9 \times max\_w_{182}).$$

The triggering event of this trigger is that a new weight reading of the user is entered into his PHR. The applicable condition of this trigger is that the user has COPD. Each time a new weight reading on the COPD patient is recorded in his PHR, this trigger is checked to see whether its firing condition is satisfied. If so, the abnormal event of weight loss in the COPD patient is detected. Subsequently, iPHR pushes the related healthcare information mentioned above to him.

Scenario 2: Unintentional weight gain caused by psychotropic medication usage

As is widely known, unintentional weight gain can lead to serious health issues. Weight gain of 7% or more is considered clinically significant [44]. The use of most psychotropic medications, such as lithium and valproic acid, can cause unintentional weight gain. It would be desirable to keep track of the weight of a patient who is taking such medication. When significant weight gain is detected in the past 12 months, it would be beneficial for iPHR to provide him with related healthcare information on how to lose weight appropriately.

In this scenario, we consider a patient who is taking lithium. We define a trigger for the abnormal event of unintentional weight gain caused by psychotropic medication usage in a way similar to that described in "Scenario 1: Weight loss in COPD patients". More specifically, the minimum weight recorded on the user in the last n days is

$$min\_w_n = \min_{d_i \geq D-n} w_i.$$

The firing condition of this trigger is

$$W \geq 1.07 \times min\_w_{365},$$

reflecting that clinically significant weight gain is detected in the past 12 months. The triggering event of this trigger is that a new weight reading of the user is entered into his PHR. The applicable condition of this trigger is that the user is taking a psychotropic medication that can cause unintentional weight gain.

Design of triggers and monitoring in iPHR

In this section, we present the design of triggers and monitoring in iPHR. The overall workflow of triggers and monitoring in iPHR is shown in Fig. 1 and consists of five steps. In step 1, the user's medical data is collected from multiple sources. In step 2, the user's medical data is preprocessed. Artifacts are filtered out and essential information is extracted as various features. In step 3, abnormal events that may have potential medical impact are detected from the user's medical data using triggers. In step 4, personalized healthcare information related to the detected abnormal events is pushed to the user in its basic form using natural language generation technology. In step 5, more detailed healthcare information related to the detected abnormal events is retrieved from the Web and presented to the user.

Step 1: Collecting the user's medical data

iPHR collects the user's medical data from various sources, such as wearable sensors [15] and other personal medical monitoring devices, in a way similar to that described in [16]. This data is recorded in the user's PHR and includes:

(1) Physiological signals: Continuous time series data extracted from physiological sensors, such as electrocardiogram, electroencephalogram, pulse rate, and oxygen saturation level (SpO2).

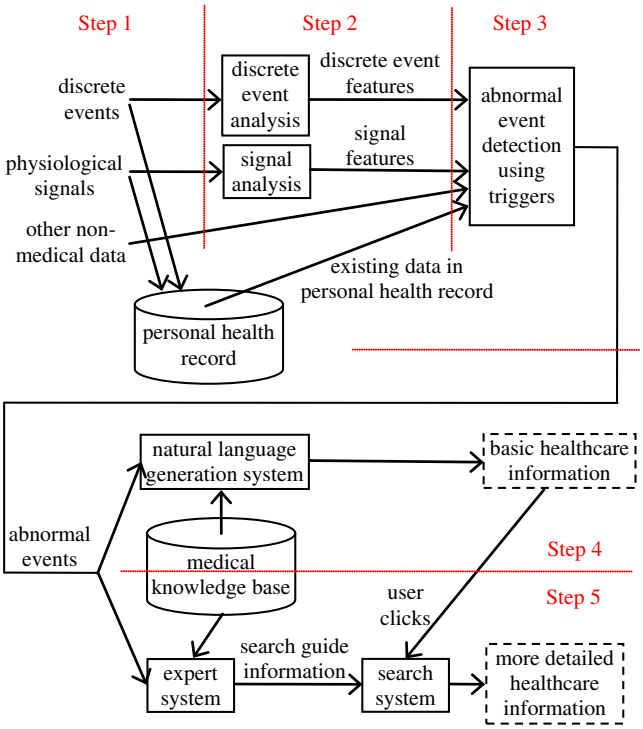

Fig. 1 Workflow of triggers and monitoring in iPHR

(2) Discrete events: Actions taken, observations made, and discrete data items that are measured sporadically, such as blood pressure, blood glucose level, and body weight.

When iPHR runs out of storage space, it may choose to delete some old physiological signals that occupy extensive storage space.

iPHR also collects other non-medical data, including weather data and environmental parameters, that may have some impact on the user's health condition. For instance, asthma patients should avoid going to areas where air quality is bad. Air quality data can be captured by (portable) air quality monitors.

iPHR handles the non-medical data and the user's medical data in a similar way. In the rest of the paper, we focus on the user's medical data.

The data collected by iPHR is of two types [16]:

(1) **Silent**: Data recording is automated and involves no user interaction. For example, many wearable sensors [15] can connect to the computer hosting iPHR via a programming interface.
(2) **Non-silent**: Users need to interact with iPHR to record the data, such as food intake.

Step 2: Pre-processing the user's medical data

There are two sub-steps in pre-processing the user's medical data. In the first sub-step, artifacts and incorrectly entered data are filtered out. In the second sub-step, essential information is extracted as various features.

Sub-step 2.1: Filtering out artifacts and incorrectly entered data

In general, there are two types of invalid data: artifacts in silent data and incorrectly entered data items in non-silent data. The purpose of the first sub-step is to detect and remove/correct invalid data.

Artifacts are typically segments of automatically recorded, physiological signals that correspond to no actual values. They can arise from multiple factors, such as a wearable sensor becomes detached from the user's body. We use the same methods as those used in [18] to detect and remove artifacts. For example, we perform range checking to identify physiologically implausible values.

From time to time, users may enter data incorrectly, e.g., by making typos or forgetting to enter data items. We use the same data validation techniques as those used in [38], such as range checking, to detect incorrectly entered data. Once such data is detected, we ask the user to reenter the corresponding data items.

Sub-step 2.2: Extracting essential information

After filtering out invalid data, we extract various features from the user's medical data based on the firing conditions of the triggers that will be used in step 3. These features capture the essential information of the user's medical data. For instance, for Scenario 1 described in "Scenario 1: Weight loss in COPD patients", the features extracted from the weight reading of the COPD patient are W, max_w_{30}, and max_w_{182}.

Typically, features of physiological signals, such as heart rate variability of electrocardiogram, are extracted using standard medical signal processing techniques [45]. The computation of a feature involves data that comes from one or more sources. In the simplest case, a feature can be the original data item itself.

Step 3: Detecting abnormal events

iPHR uses triggers and features extracted from the user's medical data to detect abnormal events that may have potential medical impact. Based on the user's medical condition, iPHR automatically determines which triggers will be used for abnormal event detection.

Compiling triggers

iPHR's knowledge base stores a set of triggers pre-compiled by medical professionals. Each trigger corresponds to an

abnormal event. Since compiling triggers requires extensive medical knowledge, average iPHR users are not involved in the trigger compilation process.

Due to the large number of potential health problems, there are numerous types of abnormal events. To have comprehensive coverage of them, many triggers are required to be compiled. In general, there are two approaches for compiling triggers [21]:

(1) In the knowledge-intensive approach, medical professionals compile triggers by using their medical knowledge and checking medical references. This approach requires medical professionals to allocate significant time.
(2) In the data-intensive approach, we collect users' medical data and use machine learning techniques to automatically learn triggers, such as their firing conditions. Then medical professionals use their medical knowledge to judge whether the learned triggers are appropriate. This approach requires collecting a large amount of (possibly labeled) medical data from users.

Regardless of which approach is used, trigger compilation is a challenging and time-consuming task. Our strategy for performing this task is to start from a subset of abnormal events that are both important and relatively easy to handle, and gradually expand the scope to other abnormal events. In this way, we can have an active iPHR system running once the corresponding triggers are compiled. For future work, we are interested in investigating whether any semi-automatic method can be used to facilitate this compilation task [55]. One possible approach is to develop a protocol for medical professionals to follow in compiling triggers. Another possible approach is to discover patterns from already compiled triggers and use these patterns to help compile more triggers.

Active trigger pool

In iPHR's knowledge base, many triggers are stored, whereas only a subset of them applies to a particular user. For each stored trigger, we use its applicable condition to decide whether it applies to the user. This is often done with the help of some medical ontologies. For example, if the applicable condition of a trigger is that the user has lung disease, then this trigger also applies to a user who has COPD, as COPD is one kind of lung disease. To quickly find all triggers that apply to the user, the stored triggers are indexed based on the "atomic items" appearing in their applicable conditions. For instance, if the applicable condition of a trigger is that the user has either COPD or asthma, then this trigger is indexed in both the entry for COPD and the entry for asthma. All stored triggers that apply to the user are automatically copied into an active trigger pool. Only these triggers are used to detect abnormal events.

The active trigger pool keeps only the most specific triggers. Intuitively, a trigger T_1 is more specific than another trigger T_2 if both triggers fire under the same condition on the user's medical data whereas T_1 targets the user's medical condition better than T_2. More specifically, T_1 and T_2 have the same firing condition and the same triggering event. T_1's applicable condition (e.g., the user has COPD) implies T_2's applicable condition (e.g., the user has lung disease). Such implication is often determined by using some medical ontologies (e.g., to check whether one disease is more specific than another disease). When both T_1 and T_2 appear in the active trigger pool, T_2 will be removed from the active trigger pool.

When the medical condition of the user changes, such as she is diagnosed with a new disease or she starts a new medication, iPHR automatically maintains the active trigger pool by adding new triggers that apply to her now and/or removing existing triggers that no longer apply to her. In case a trigger T_1 is removed from the active trigger pool, we check whether any trigger T_2 was previously removed from the active trigger pool because T_2 is less specific than T_1. If we find such a T_2 that still applies to the user now and is not less specific than any other trigger in the active trigger pool, we add T_2 back to the active trigger pool.

Identifying related triggers and computing the corresponding features

Typically, every kind of physiological signal, discrete event, or non-medical data corresponds to a different data source (see "Step 1: Collecting the user's medical data"). Whenever a new piece of data comes from a data source, we need to identify which triggers in the active trigger pool are related to this data source and check their firing conditions. To facilitate this identification process, we maintain a linkage data structure that links data sources, features, and the triggers in the active trigger pool together. More specifically, each data source links to all features that are extracted from the data from this data source. Each trigger in the active trigger pool links to all features used in its firing condition. All links used in this linkage data structure are bidirectional. In this way, given a feature, we can easily find all data sources linked to it as well as all triggers linked to it.

Figure 2 shows an example of this linkage data structure, where:

(1) Triggers T_1, T_2, and T_3 are in the active trigger pool.
(2) Features F_1 and F_2 are extracted from the data that comes from the data source S_1.

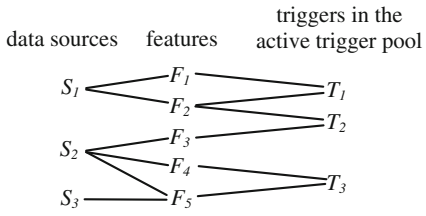

Fig. 2 An example linkage data structure linking data sources, features, and the triggers in the active trigger pool together

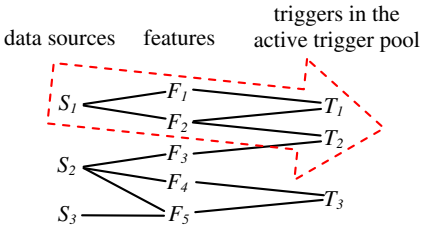

Fig. 3 Using the linkage data structure shown in Fig. 2 to find all triggers that are in the active trigger pool and related to the data source S_1

(3) Features F_3 and F_4 are extracted from the data that comes from the data source S_2.
(4) Feature F_5 is extracted from both the data coming from S_2 and the data coming from the data source S_3.
(5) The firing condition of T_1 uses F_1 and F_2.
(6) The firing condition of T_2 uses F_2 and F_3.
(7) The firing condition of T_3 uses F_4 and F_5.

When a new piece of data comes from a data source S, we first find all features linked to S. Then for each such feature, we find all triggers linked to it. By going through the links in the linkage data structure from left to right in this way, we find all triggers that are in the active trigger pool and related to S.

Next, to check the firing conditions of these triggers, we go through the links in the linkage data structure from right to left to find all features that need to be computed as well as their corresponding data sources. More specifically, for each of these triggers, we find all features linked to it. Then for each such feature, we find all data sources linked to it and compute its value using the data from these data sources.

For instance, consider the linkage data structure shown in Fig. 2. Suppose a new piece of data comes from the data source S_1. By going through the links in the linkage data structure from left to right as shown in Fig. 3, we find that both triggers T_1 and T_2 are related to S_1 and need to check their firing conditions. By going through the links in the linkage data structure from right to left as shown in Fig. 4, we find that this checking requires computing the features F_1, F_2, and F_3 by using both the data from S_1 and the data from the data source S_2. More specifically, computing F_1 (F_2) requires the new piece of data from S_1 and maybe some old data that previously came from S_1 and has already been stored in the user's PHR. Computing F_3 requires some old data that previously came from S_2 and has already been stored in the user's PHR.

Reducing feature computation overhead

Usually feature computation accounts for most of the overhead of checking the related triggers' firing conditions. To reduce this overhead, we borrow four techniques from the database field.

First, for some features, such as certain statistics over a sliding window, we can maintain some auxiliary data structures to facilitate their computation [12]. For a particular feature, if its precise value is expensive to compute, we may choose to use some approximation techniques to compute an approximate value of it [12].

Second, to avoid repeated computation as much as possible, it could be beneficial to cache the most recently computed value of each feature [23], whereas the exact amount of saved computation depends on both the definition and the concrete usage scenario of this feature. For example, consider the linkage data structure shown in Fig. 2. When a new piece of data comes from the data source S_1, the firing condition of the trigger T_2 needs to be checked. In this case, there is no need to re-compute the feature F_3 used in that firing condition if F_3's value remains the same as when it was computed last time, i.e., upon the most recent arrival of a new piece of data from the data source S_2.

Third, some optimization can be performed to avoid computing certain features used in the firing condition of a trigger. For example, consider the trigger that is defined in "Scenario 1: Weight loss in COPD patients" and has the firing condition

$$(W < 0.95 \times max\_w_{30}) \vee (W < 0.9 \times max\_w_{182}).$$

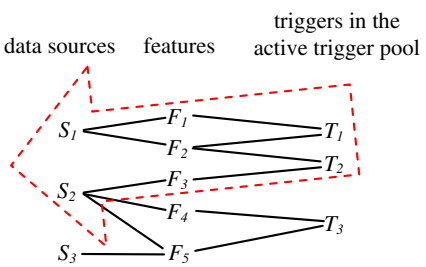

Fig. 4 Using the linkage data structure shown in Fig. 2 to find all features that need to be computed as well as their corresponding data sources

When the first sub-condition $W<0.95\times max\_w_{30}$ is satisfied, the entire firing condition is satisfied and hence there is no need to compute the feature max_w_{182} used in the second sub-condition $W<0.9\times max\_w_{182}$.

The feature max_w_{30} is less expensive to compute than the feature max_w_{182}, although various sub-conditions in the firing condition have different selectivities. To reduce the overall cost of checking a firing condition C_f, we need to determine a good order of evaluating C_f's sub-conditions based on both their selectivities and the computational costs of the features used in them. This can be done in a way similar to that of optimizing the evaluation of a database query that has expensive predicates [24]. Moreover, if the firing conditions of multiple triggers use some common features and need to be checked simultaneously, we can borrow multi-query optimization techniques [43] from the database field to reduce the overall cost of checking these firing conditions.

Fourth, similar to that in multi-query optimization [43] and stream computing [22], we can share the computation of the common parts of multiple features to reduce the total cost of computing these features. The general approach is to break the computation of a feature into the computation of multiple sub-features or multiple levels of sub-features and then let features share sub-features.

For instance, consider the scenario shown in Fig. 5, where:

(1) The computation of the feature F_1 is broken into the computation of the sub-features SF_1 and SF_2.
(2) The computation of the feature F_2 is broken into the computation of the sub-features SF_2 and SF_3.
(3) Both SF_1 and SF_2 are extracted from the data that comes from the data source S_1.
(4) SF_3 is extracted from the data that comes from the data source S_2.
(5) The firing condition of the trigger T_1 uses F_1.
(6) The firing condition of the trigger T_2 uses F_2.

When a new piece of data comes from the data source S_1, the feature F_1 needs to be computed to check the firing condition of the trigger T_1. Simultaneously, the feature F_2 needs to be computed to check the firing condition of the trigger T_2. In this case, the computation of the sub-feature SF_2 needs to be done only once and then can be shared by F_1 and F_2.

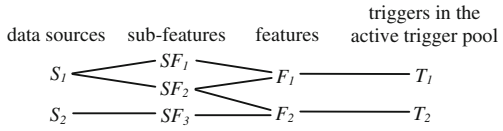

Fig. 5 An example of sharing the computation of the common parts of multiple features using one level of sub-features

Checking the related triggers' firing conditions and firing triggers

After identifying the related triggers in the active trigger pool and computing their corresponding features, their firing conditions are checked. For each such trigger, if its firing condition is satisfied, it fires and its action is taken. At the very least, this action includes reporting the abnormal event corresponding to this trigger (and often how this trigger's firing condition is satisfied) to the natural language generation system of iPHR. For most triggers, such reporting is the sole activity of their actions.

If the action of one trigger can affect either the firing condition or the triggering event of another trigger, then in theory recursive trigger firing is possible. In this case, the sequence of checking the related triggers' firing conditions is important [36]. An example of recursive trigger firing is as follows. When a trigger T_1 fires, its action causes the firing condition of another trigger T_2 to be satisfied. Then T_2 fires. T_2's action leads to the satisfaction of T_1's firing condition and subsequently the firing of T_1 again.

In practice, our trigger design disallows recursive trigger firing. iPHR uses triggers to detect and report abnormal events. Upon the arrival of new data, it is meaningless to report any particular abnormal event more than once. To control the sequence of checking the related triggers' firing conditions, medical professionals could at trigger compilation time assign each trigger a priority value that reflects the priority of checking its firing condition.

In most cases, for any trigger in the active trigger pool, its action affects neither the firing condition nor the triggering event of any other trigger in the active trigger pool. In this case, the sequence of checking the related triggers' firing conditions has no impact on the set of detected abnormal events. Consequently, the related triggers can have their firing conditions checked and subsequently fire in any order or in parallel.

Sometimes it is desirable to avoid repeatedly firing the same trigger within a short period of time. Upon the first time that a trigger fires, iPHR pushes healthcare information related to the corresponding abnormal event to the user. There is usually little gain on pushing the same information to the user again within a short period of time, unless this abnormal event is critical and deserves extra attention from the user.

For instance, consider the trigger that is defined in "Scenario 1: Weight loss in COPD patients". Suppose today weight loss is detected for the COPD patient based on his weight readings in the past 30 days, i.e., he has lost >5% of his weight in the past month. Then as shown in Fig. 6 where each dot represents a weight reading, it is possible

that weight loss will be detected for him again tomorrow based on his weight readings in the previous 30 days because weight gain may not occur immediately. After pushing healthcare information related to weight loss in COPD patients to him today, it would be redundant to push this same information to him tomorrow.

One approach to avoid repeatedly firing the same trigger within a short period of time is to associate each trigger with a pre-determined rest time period P. P is measured either in the amount of physical time (e.g., 30 days) or in the number of data items that come from a particular data source related to this trigger. Medical professionals specify P at the time when they compile this trigger. Once this trigger fires, we let it rest for P if the value of P is not null. In other words, we stop firing it for P if the value of P is not null.

During the period that a trigger is resting, we may reduce feature computation overhead by avoiding computing certain features linked to this trigger unnecessarily. For instance, consider the linkage data structure shown in Fig. 2. Suppose the trigger T_2 is resting. Then when a new piece of data comes from the data source S_1, we need to compute the features F_1 and F_2 but not the feature F_3, because F_3 is needed only by T_2.

Step 4: Pushing personalized healthcare information to the user in its basic form

When one or more abnormal events are detected from the user's medical data and their corresponding triggers fire, they are collected together and reported to the natural language generation system of iPHR. Then iPHR uses medical knowledge and natural language generation technology [41] to generate basic personalized healthcare information related to these abnormal events.

Once generated, the healthcare information is pushed to the user in one of multiple ways:

(1) If the newly arrived data leading to the detected abnormal events is non-silent (user-entered), iPHR can display the healthcare information on the confirmation Web page when the user clicks the submit button to enter her data into her PHR. The top section of the confirmation Web page will indicate that her data has been successfully entered. The healthcare information follows that top section.

(2) If the newly arrived data leading to the detected abnormal events is silent, we can proceed in one or more of the following ways:

   (a) iPHR can display the healthcare information on its main Web page for the user to see during login.
   (b) iPHR can send the healthcare information to the user in an email.
   (c) iPHR can send a summary of the healthcare information to the user in a text message.

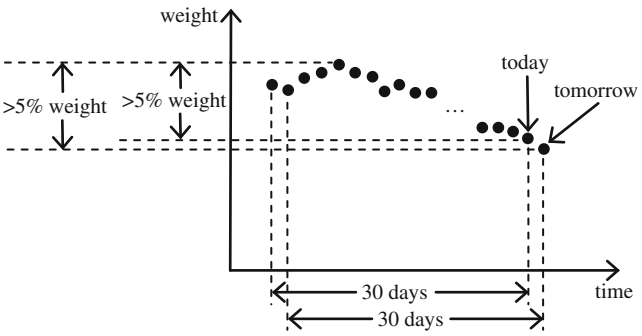

Fig. 6 An example of repeatedly detecting weight loss in a COPD patient within a short period of time

In the rest of the paper, we focus on the method of displaying the healthcare information on the confirmation Web page. Other methods of displaying the healthcare information can be handled in a similar manner.

Using templates to generate basic personalized healthcare information

iPHR uses templates to generate basic personalized healthcare information. At trigger compilation time, for each abnormal event $E_{abnormal}$, medical professionals compile a template of basic healthcare information related to it as well as assign it an importance value reflecting its importance or urgency. Both this template and this importance value are stored in iPHR's knowledge base together with the trigger T corresponding to $E_{abnormal}$.

This template of basic healthcare information has the following four components:

(1) **Component C_1**: A high-level description of the abnormal event $E_{abnormal}$, e.g., its name.
(2) **Component C_2**: How $E_{abnormal}$ is detected for the user. For instance, which part of the trigger T's firing condition is satisfied. Knowing this can both help the user better understand $E_{abnormal}$ and facilitate her to report $E_{abnormal}$ to her physician if necessary.
(3) **Component C_3**: The potential health risks associated with $E_{abnormal}$.
(4) **Component C_4**: When $E_{abnormal}$ occurs, what interventions can be performed to help improve the user's condition and/or reduce the potential health risks listed in the component C_3. Example interventions include home nursing activities [28] and the use of home medical products [30]. The interventions that patients and caregivers can perform at home or in the

community are explicitly marked and separated from the interventions that are typically performed in the hospital.

For certain abnormal events, one or more but not all of these four components could be empty. The content of the components C_1, C_3, and C_4 is pre-compiled by medical professionals. If there is only one way for the trigger T to get fired, the content of the component C_2 is pre-compiled by medical professionals. Otherwise, if there is more than one way for T to get fired, the content of C_2 is dynamically generated based on how T gets fired.

One method of dynamically generating the content of the component C_2 is as follows. For each possible way of satisfying the trigger T's firing condition, medical professionals pre-compile some text describing it and store this text in iPHR's knowledge base. When T's firing condition is satisfied in this way, this text is used as the content of C_2. Recall that the abnormal event detection system of iPHR reports how a trigger's firing condition is satisfied.

For instance, consider the abnormal event of weight loss in the COPD patient that is described in "Scenario 1: Weight loss in COPD patients". The corresponding trigger's firing condition is

$$(W < 0.95 \times max\_w_{30}) \vee (W < 0.9 \times max\_w_{182}).$$

The text pre-compiled for the first sub-condition $W<0.95\times max\_w_{30}$ is "You have lost >5% of your weight in the past month." The text pre-compiled for the second sub-condition $W<0.9\times max\_w_{182}$ is "You have lost >10% of your weight in the past six months."

Some special care is needed with the interventions included in the content of the component C_4. An iPHR user often has multiple health issues [48]. An intervention may be generally suitable for handling one abnormal event, but become undesirable in the presence of another health issue. In medicine, contraindication is the general term describing this phenomenon and refers to a health issue that makes a particular intervention undesirable [5]. Ideally, based on the user's health condition, iPHR should dynamically remove contraindicated interventions (at least the absolutely contraindicated ones that should be performed under no circumstance [5]) from the content of C_4. For this purpose, we use a method similar to that described in [29] for identifying contraindicated home nursing activities. The high-level idea is to link the collected contraindication information with the nodes in the medical ontology of International Classification of Diseases (ICD-10) [53]. Contraindicated home nursing activities are identified by navigating the ICD-10 hierarchy.

When the content of the four components is populated, iPHR uses standard natural language generation techniques [41] to combine them together in a sequence to form the basic personalized healthcare information related to the abnormal event. This combination follows a relatively fixed format and is usually performed using a set of rules predefined jointly by medical professionals and the builders of iPHR.

The case with multiple concurrent abnormal events

The above discussion is based on only one trigger firing at a time. In general, multiple triggers can fire at a time. Consequently, multiple abnormal events are reported to the natural language generation system of iPHR simultaneously. In this case, for each of these abnormal events, we use the above-mentioned method to generate a piece of basic personalized healthcare information related to it. Then we sort all these pieces of information in descending order of the importance values of their corresponding abnormal events. Recall that at trigger compilation time, medical professionals assign an importance value to each abnormal event. Finally, we put all these pieces of information one after another to form the basic personalized healthcare information that will be pushed to the user.

There is one complexity with the case of multiple concurrent abnormal events. Consider the following two interventions. The first intervention is included in the content of the component C_4 for one abnormal event. The second intervention is included in the content of C_4 for another abnormal event. These two interventions may interact with one another (e.g., drug-drug interaction [46]) and hence cannot be performed simultaneously. In this case, it is desirable for iPHR to automatically identify and mark the conflicting interventions in the content of C_4 so that users can become aware of these conflicts [2]. To achieve this goal, we use a method similar to that for automatically checking adverse drug interactions [46].

Step 5: Presenting more detailed healthcare information to the user

Once generated, the basic personalized healthcare information related to the detected abnormal events is presented to the user as a Web page. After viewing this Web page, the user is often interested in knowing more details about one or more of the specific items in this basic information. Many users would want a detailed explanation of a potential health risk associated with a detected abnormal event as well as detailed implementation procedures for a home nursing activity. To serve this purpose, iPHR automatically adds hyperlinks into this Web page, in a way similar to that described in [28]. By

clicking these hyperlinks, the user obtains detailed information of the corresponding items. Due to a lack of medical knowledge, the user may not know that detailed information of certain items exists, but she will realize this once she sees these hyperlinks.

At a high level, our method of presenting more detailed healthcare information to the user works as follows. Consider an abnormal event $E_{abnormal}$. At trigger compilation time, for the content included in the template of basic healthcare information related to $E_{abnormal}$, medical professionals use their medical knowledge to mark one or more items that they anticipate some iPHR users would want to know more about. For each such item, a hyperlink is added. When the user views the basic personalized healthcare information related to the detected abnormal events, if he clicks this particular hyperlink, iPHR will first use medical knowledge and expert system technology to automatically form the corresponding queries. These queries and Web search technology will then be used to retrieve detailed information about the item. This retrieved information is then presented to the user.

Similar to that described in [30], the basic idea of our method of obtaining detailed information about a particular item I is as follows. At trigger compilation time, medical professionals compile a set S of phrases for I as its so-called search guide information and store S in iPHR's knowledge base. Each phrase in S provides one way of retrieving detailed information about I. The search results for all phrases in S are combined together in an appropriate order to become the retrieved detailed information about I.

Search guide phrase compilation is a time-consuming task. For future work, we are interested in investigating whether any semi-automatic method can help speed up this compilation process [55]. One possible approach is to proceed as follows. First, the healthcare professional provides a search guide phrase for the item I as a seed. Then an automatic tool is used to submit this phrase as a query to a large-scale Web search engine such as Google [17], retrieve back one or more Web pages about I, analyze them, and extract search guide phrase candidates from them. Finally, the medical professional checks these candidates to determine which ones are good to serve as search guide phrases.

Next, we present the details of our method of obtaining detailed information about the item I. For each phrase Q in the set S of search guide information, we submit it as a query to a large-scale Web search engine such as Google [17] to retrieve a set R_Q of k Web pages. In our current implementation of iPHR, the default value of k is 100. The retrieval is performed using the API of the Web search engine [19, 51]. All retrieved Web pages are merged into a set $R_{all} = \bigcup_{Q \in S} R_Q$. Then these Web pages are ranked and presented to the user as detailed information about I.

Ranking web pages

To rank the Web pages in the set R_{all}, we apply a method similar to that used in [30]. We use a conceptual query $Q_c = \vee_{Q \in S} Q$, which combines all phrases in the set S of search guide information together, to represent the item I. For each Web page $P \in R_{all}$, we compute a relevance score $score_P$ according to which P is ranked.

We start from the following probability computation:

$$p(P|I) = p(P|Q_c) = p(P,\ Q_c)/p(Q_c) \propto p(P,\ Q_c).$$

Ignoring the second- and higher- order terms, we have

$$p(P,\ Q_c) = p\left(P,\ \vee_{Q \in S} Q\right) \approx \sum\nolimits_{Q \in S} p(P,\ Q).$$

Usually a large-scale Web search engine uses $p(P,\ Q) = p(Q|P)p(P)$ to rank Web pages retrieved for the query Q. As the prior probability that the Web page P is relevant to a query, $p(P)$ is often computed via some link analysis method such as PageRank [34]. As the conditional probability of producing Q given P, $p(Q|P)$ is computed using the anchor text of links to P [7], the text on P, and some retrieval model such as the language modeling approach [35].

In our case of iPHR, the ranking score $p(P, Q)$ computed by the large-scale Web search engine is unavailable to us. To overcome this difficulty, as a rough approximation we assume that this ranking score is inversely proportional to $rank(P, Q)$, the rank of the Web page P among all Web pages retrieved by the phrase Q. That is,

$$p(P,\ Q) \propto 1/rank(P,\ Q)\,.$$

This assumption reflects the fact that $p(P, Q)$ decreases as $rank(P, Q)$ becomes larger. If P is in the set R_Q of top-k Web pages retrieved by Q, $rank(P, Q)$ is available to us. Otherwise, if $P \notin R_Q$, we take $rank(P, Q)$ to be infinite so that $p(P, Q)=0$. Then we have

$$\begin{aligned} p(P|I) &\propto p(P,\ Q_c) \\ &\approx \textstyle\sum_{Q \in S} p(P,\ Q) \\ &\propto \textstyle\sum_{Q \in S,\ P \in R_Q} 1/rank(P,\ Q)\,. \end{aligned}$$

Consequently, the relevance score $score_P$ is computed as $\sum_{Q \in S,\ P \in R_Q} 1/rank(P,\ Q)$.

For each Web page $P \in R_{all}$, its title, URL, and snippet S_P (i.e., some words extracted from P) are presented to the user. One way to obtain S_P is to use the method described

in [52] by treating the combination of all phrases in the set S of search guide information as the query. Alternatively, for the phrase that is in S and ranks P the highest, the snippet returned by the Web search engine can serve as S_P. More specifically, consider every phrase $Q \in S$ such that $P \in R_Q$. Since $P \in R_{all}$, there is at least one such phrase. Among these phrases, we select the one that has the smallest $rank(P, Q)$ and hence ranks P the highest. Intuitively, P is likely to be most relevant for this phrase. When this phrase is submitted as a query to retrieve P, the Web search engine returns a snippet of P. This snippet serves as S_P.

Retrieving home medical products

The above discussion does not address the case that the item I refers to a list of home medical products (HMPs). In the content of the component C_4 of the template of basic healthcare information related to an abnormal event, some items could refer to a list of HMPs. Typically, such an item includes a hyperlink and is written in a way like "Click here to view home medical products related to a topic t." If the user clicks this hyperlink, he will be presented with a list of HMPs related to t.

This list L of HMPs is obtained using a method similar to that described in [30]. The high-level idea is as follows. Medical professionals pre-compile a list of keyword phrases as the HMP search guide information of the topic t. Using a vertical search engine specifically designed for HMPs, each such phrase retrieves some HMPs related to t. The retrieved HMPs for all these phrases are combined together. Then after removing contraindicated HMPs using the method described in [29], the remaining HMPs are ranked in an appropriate order to become L.

Experimental results

At the time of this writing, we have implemented the general framework of the active iPHR system and compiled a few triggers. To give the reader a feeling of the active iPHR system, we present detailed results on the healthcare information that the active iPHR system provides for the two scenarios described in "Motivation for triggers and monitoring in iPHR". The reader is encouraged to browse all Web pages listed in this section to obtain a good understanding of these detailed results.

Detailed results for Scenario 1

Scenario 1 described in "Scenario 1: Weight loss in COPD patients" is about weight loss in COPD patients. The corresponding trigger's firing condition is

$$(W < 0.95 \times max\_w_{30}) \vee (W < 0.9 \times max\_w_{182}).$$

When the first sub-condition $W<0.95\times max\_w_{30}$ is satisfied, iPHR pushes some basic personalized healthcare information to the user as shown in Fig. 7. As indicated, we mark the four components C_1, C_2, C_3, and C_4 of the template from which this basic healthcare information is generated.

For the "nutritional therapy" item in the component C_4, its pre-compiled search guide information contains three phrases: *COPD nutritional therapy*, *COPD nutritional supplement*, and *COPD nutrition*. If the user clicks this item, iPHR will display detailed information about the item as shown in Table 1.

The item that is in the component C_4 and identified as "Click here to view related food and nutritional supplements" refers to a list of HMPs. The HMP search guide information pre-compiled for this item contains many phrases, such as *N-acetylcysteine*, *L-carnitine*, and *bromelain*. If the user clicks this item, iPHR will display various food and nutritional supplements retrieved by these phrases.

For the "exercise" item in the component C_4, its pre-compiled search guide information contains one phrase: *COPD exercise*. If the user clicks this item, iPHR will display detailed information about the item as shown in Table 2.

Detailed results for Scenario 2

Scenario 2 described in "Scenario 2: Unintentional weight gain caused by psychotropic medication usage" is about

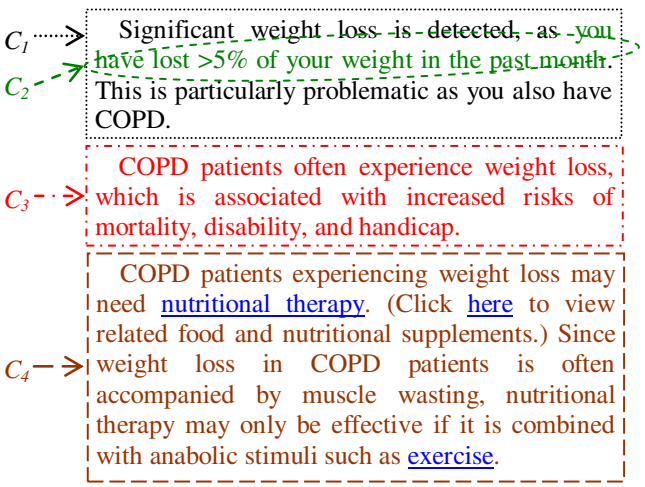

Fig. 7 An example of basic personalized healthcare information that iPHR provides for Scenario 1

Table 1 A subset of retrieved detailed information about the "nutritional therapy" item for Scenario 1

rank	URL	topic
1	http://my.clevelandclinic.org/disorders/chronic_obstructive_pulmonary_disease_copd/hic_nutritional_guidelines_for_people_with_copd.aspx	nutritional guidelines for people with COPD
2	http://www.todaysdietitian.com/newarchives/td_020909p54.shtml	nutrition and COPD - dietary considerations for better breathing
3	http://www.lef.org/protocols/respiratory/copd_01.htm	nutritional therapy for COPD

unintentional weight gain caused by psychotropic medication usage. When the corresponding trigger's firing condition is satisfied, iPHR pushes some basic personalized healthcare information to the user as shown in Fig. 8. As indicated, we mark the four components C_1, C_2, C_3, and C_4 of the template from which this basic healthcare information is generated.

For the "exercise" item in the component C_4, its precompiled search guide information contains one phrase: *psychotropic medication weight gain exercise*. If the user clicks this item, iPHR will display detailed information about the item as shown in Table 3.

For the "healthy diet" item in the component C_4, its precompiled search guide information contains two phrases: *psychotropic medication lose weight healthy diet* and *psychotropic medication weight gain dietary guidance*. If the user clicks this item, iPHR will display detailed information about the item as shown in Table 4.

System response time

We measured the response time of the active iPHR system on a computer with two 3 GHz processors, 2 GB memory, and one 111 GB disk. For each tested scenario, in response to the arrival of a new piece of the user's medical data that led to the detection of one or more abnormal events, the active iPHR system consistently provided basic personalized healthcare information in less than one second.

Usually, the number of triggers applicable to a particular health issue is not large. A typical user has a small number of health issues, if any. Consequently, the number of triggers in the active trigger pool is usually not large. We expect the active iPHR system to quickly provide basic personalized healthcare information in most cases, unless the active trigger pool has a trigger whose firing condition is extremely computationally expensive to check.

In our testing, when the user clicked a hyperlink embedded in the basic personalized healthcare information, the active iPHR system provided the corresponding, detailed healthcare information in less than one second. This excludes the time spent on retrieving Web pages from the large-scale Web search engine using its API. The retrieval of Web pages can be implemented more efficiently [54] if iPHR is tightly integrated with the Web search engine, as is the case with the PHR systems of Microsoft [32] and WebMD [31] where Web pages can be retrieved locally from the Web search engine.

Related work and discussions

Health information systems

The National Library of Medicine (NLM) is building a PHR system [33]. If the user clicks the name of a recorded medication or disease, the NLM PHR will display MedlinePlus information resources about this medication or disease. The NLM PHR can generate reminders that preventive care interventions are due, e.g., it is time for the annual flu shot.

Cimino has built an Infobutton Manager for electronic medical records [8]. For each medical concept appearing in

Table 2 A subset of retrieved detailed information about the "exercise" item for Scenario 1

rank	URL	topic
1	http://www.webmd.com/lung/copd/copd-and-exercise-breathing-and-exercise-programs-for-copd	COPD and exercise: breathing and exercise programs for COPD
2	http://my.clevelandclinic.org/disorders/chronic_obstructive_pulmonary_disease_copd/hic_copd_exercise_and_activity_guidelines.aspx	COPD exercise and activity guidelines
3	http://my.clevelandclinic.org/disorders/chronic_obstructive_pulmonary_disease_copd/hic_copd_exercise_precautions.aspx	COPD exercise precautions

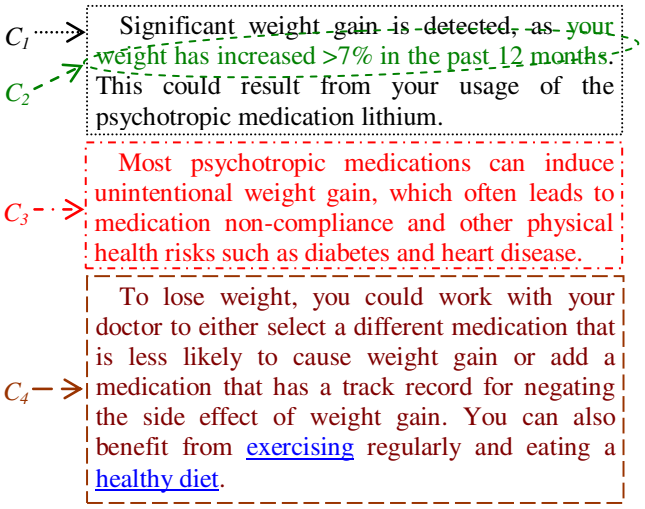

Fig. 8 An example of basic personalized healthcare information that iPHR provides for Scenario 2

the electronic medical record, the Infobutton Manager provides a fixed set of questions that physicians ask most often and uses manually pre-constructed queries to retrieve answers to these questions from certain resources in real time. Each answer is retrieved using one query.

In several telemedicine systems for managing outpatients with chronic diseases [21, 38], alarms are sent to healthcare professionals when potential health problems are detected from the user's medical data. In practice, many potential health problems are minor and do not require immediate attention from inundated healthcare professionals. It could be sufficient for consumers to receive timely education and then possibly take some preventative actions themselves. Our active iPHR system can serve this purpose by actively pushing related healthcare information to the user. In general, the more informed the user is, the more willing he is to perform actions to help improve his condition. Moreover, in our active iPHR system, we can define triggers for abnormal events that may have potentially severe and possibly immediate medical impact. The actions of these triggers are to use the methods described in [21, 38] to send alarms to healthcare professionals and/or the user's caregivers.

Apiletti *et al.* [1] proposed a framework for monitoring the patient's health condition via real-time analysis of physiological data. Given historical physiological data, this framework automatically learns detection models tailored to specific conditions, such as a particular disease or a specific patient. When high-risk situations are detected, alerts are triggered, but no healthcare information related to these alerts is provided.

Clinical natural language generation systems

Gatt *et al.* [18, 37] have built a natural language generation system for intensive care units. This system can automatically generate textual summaries from the patient's medical data, e.g., by reporting that SpO2 increased from 88% to 97% between 3:00 am and 6:00 am. Harris [20] described a natural language generation component of an electronic medical record system that generates narrative recording the doctor/patient encounter. Dalal *et al.* [11] presented an experimental system that can generate multimedia briefings about patient status.

In contrast to our active iPHR system, none of the systems mentioned above can both continuously monitor the user and automatically push related personalized healthcare information to the user.

Expert systems, recommender systems, and information retrieval

Rules are widely used in expert systems [25]. The way rules are fired in expert systems is similar to the way triggers are fired in our active iPHR system. Medical expert systems usually use rules to generate medical hypotheses. In contrast, our active iPHR system uses triggers to continuously monitor the user and automatically push related personalized healthcare information to the user.

Traditional recommender systems recommend items based on various factors, such as users' prior ratings, previous purchases, and profiles [4]. Our active iPHR system can be regarded as a continuous recommender system that goes beyond these factors and keeps recommending person-

Table 3 A subset of retrieved detailed information about the "exercise" item for Scenario 2

rank	URL	topic
1	http://www.netnutritionist.com/fa12.htm	the relationship between weight gain and medications for depression and seizures (covering exercise)
2	http://blogs.psychcentral.com/bipolar/2008/10/preventing-and-reversing-weight-gain-associated-with-psychiatric-medications/	preventing and reversing weight gain associated with psychiatric medications (covering exercise)
4	http://www.psycheducation.org/hormones/Insulin/weightgain.htm	weight gain and bipolar disorder treatment (covering exercise)

Table 4 A subset of retrieved detailed information about the "healthy diet" item for Scenario 2

rank	URL	topic
1	http://williGocrazy.org/Ch03.htm	last-resort weight loss plan (covering diet)
2	http://bipolar.about.com/u/ua/weightlosssupport/diet_tips.htm	readers respond: diet tips for losing weight in spite of psych meds
3	http://www.aacap.org/cs/root/facts_for_families/preventing_and_managing_medicationrelated_weight	preventing and managing medication-related weight gain (covering diet)

alized healthcare information using both medical knowledge and the user's medical data.

In distributed information retrieval and meta-search engines [6], search results from multiple sources for the same query are merged together. In contrast, when retrieving detailed information about an item, iPHR merges together the search results retrieved by different query phrases.

In a publish/subscribe system [39], users submit standing queries. All newly arrived text documents satisfying the condition of such a query are automatically routed to the user who submitted the query. In comparison, the standing triggers in our active iPHR system are automatically invoked based on the user's medical condition. For the abnormal events detected by these triggers, the related healthcare information is automatically pushed to the user.

Database triggers

Active database systems [36] use event-condition-action rules [14] for multiple purposes, such as maintaining the integrity of the information in the databases. Typically, an event-condition-action rule is a SQL trigger [13] of the form

On triggering event E if firing condition C do action A, meaning that when the triggering event E occurs and the firing condition C is satisfied, the action A will be taken. Example triggering events include the insertion of a new tuple into a particular relational table and the deletion of an existing tuple from a particular relational table. C is typically a SQL query. An example action is a relational table update.

The triggers in iPHR are different from SQL triggers in multiple aspects. First, iPHR triggers and SQL triggers are used for different purposes. Second, iPHR triggers and SQL triggers have different triggering events. Third, once fired, iPHR triggers and SQL triggers take completely different actions. Fourth, each iPHR trigger has an applicable condition, which does not exist in SQL triggers. Fifth, iPHR triggers are pre-compiled by medical professionals rather than by iPHR users. iPHR automatically determines which triggers will be used to detect abnormal events. In comparison, through a manual process, database users both define SQL triggers and decide which SQL triggers will be used.

Healthcare professionals

Our iPHR work falls into the area of consumer health informatics [27]. As mentioned in [29], healthcare information keeps updating rapidly and healthcare professionals cannot keep up with all of the latest changes. For example, each year many new home medical products enter the market. However, physicians usually are unaware of these products or receive little training on them. We would expect iPHR to supplement healthcare professionals' efforts in providing healthcare information. iPHR's knowledge base stores medical knowledge compiled by many healthcare professionals and its search system can discover the latest healthcare information from the Web.

However, iPHR is not intended to replace healthcare professionals. Regardless of what kinds of triggers are used in iPHR, they can never achieve absolute certainty on the user's potential health problems. Moreover, each person is different. Significant medical knowledge is often needed to determine whether a specific nugget of healthcare information is applicable to a particular person. Hence, the personalized healthcare information that iPHR pushes to the user should be used only for reference and educational purposes. For final decision making on which interventions should be performed, the user needs to consult her physician.

Conclusions

This paper introduces triggers and monitoring into iPHR so that iPHR can automatically detect when users need to be aware of relevant, personalized healthcare information and actively push this information to them. To illustrate what the new trigger function of iPHR can do, we present detailed results for a few scenarios. At present, we are in the process of compiling more triggers to increase the number of cases that can be handled by the active iPHR system.

Acknowledgments We thank Chunqiang Tang, Selena Thomas, and Jing Wang for helpful discussions.

References

1. Apiletti, D., Baralis, E., Bruno, G., et al., Real-time analysis of physiological data to support medical applications. *TITB* 13 (3):313–321, 2009.
2. American Dietetic Association. *Electronic medical records and personal health records: A call for the creation and inclusion of a nutrition dataset*. http://anhi.org/Learning/PDFs/Other/EMR-PHR_Nutrition_Dataset_Paper(6-29-09).pdf, 2009.
3. Anderson, G., and Horvath, J., The growing burden of chronic disease in America. *Public Health Rep.* 119(3):263–270, 2004.
4. Adomavicius, G., and Tuzhilin, A., Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *TKDE* 17(6):734–749, 2005.
5. Batavia, M., *Contraindications in Physical Rehabilitation: Doing No Harm*. Saunders, 2006.
6. Baeza-Yates, R. A., and Ribeiro-Neto, B. A., *Modern Information Retrieval*. ACM Press/Addison-Wesley, 1999.
7. Brin, S., and Page, L., The anatomy of a large-scale hypertextual Web search engine. *Comput. Netw.* 30(1–7):107–117, 1998.
8. Cimino, J. J., Use, usability, usefulness and impact of an Infobutton manager. *Proceedings of AMIA'06*, pp. 151–155, 2006.
9. Celli, B. R., MacNee, W., and ATS/ERS Task Force, Standards for the diagnosis and treatment of patients with COPD: a summary of the ATS/ERS position paper. *Eur. Respir. J.* 23 (6):932–946, 2004.
10. COPD prevalence & statistics. http://www.copdforums.com/copd-prevalence-statistics, 2009.
11. Dalal, M., Feiner, S., and McKeown, K., *et al*., MAGIC: an experimental system for generating multimedia briefings about post-bypass patient status. *Proceedings of AMIA'96*, pp. 684–688, 1996.
12. Datar, M., Gionis, A., Indyk, P., et al., Maintaining stream statistics over sliding windows. *SIAM J. Comput.* 31(6):1794–1813, 2002.
13. Database trigger. http://en.wikipedia.org/wiki/Database_trigger, 2011.
14. Event condition action. http://en.wikipedia.org/wiki/Event_Condition_Action, 2011.
15. Fletcher, R. R., Poh, M. Z., and Eydgahi, H., Wearable sensors: opportunities and challenges for low-cost health care. *Proceedings of EMBC'10*, pp. 1763–1766, 2010.
16. Giacomelli, P., Munaro, G., and Rosso, R., Using soft computer techniques on smart devices for monitoring chronic diseases: the CHRONIOUS case. CoRR abs/1103.3223, 2011.
17. Google homepage. http://www.google.com, 2011.
18. Gatt, A., Portet, F., Reiter, E., et al., From data to text in the neonatal intensive care unit: using NLG technology for decision support and information management. *AI Commun.* 22(3):153–186, 2009.
19. Google Web search API. http://code.google.com/apis/websearch/, 2010.
20. Harris, M. D., Building a large-scale commercial NLG system for an EMR. *Proceedings of INLG'08*, pp. 157–160, 2008.
21. Hudson, D. L., and Cohen, M. E., Diagnostic models based on personalized analysis of trends (PAT). *TITB* 14(4):941–948, 2010.
22. Andrade, H., Gedik, B., Wu, K.-L., et al., Processing high data rate streams in System S. *J. Parallel Distrib. Comput.* 71(2):145–156, 2011.
23. Hellerstein, J. M., and Naughton, J. F., Query execution techniques for caching expensive methods. *Proceedings of SIGMOD'96*, pp. 423–434, 1996.
24. Hellerstein, J. M., and Stonebraker, M., Predicate migration: optimizing queries with expensive predicates. *Proceedings of SIGMOD'93*, pp. 267–276, 1993.
25. Jackson, P., *Introduction to Expert Systems*, 3rd ed. Addison Wesley, 1998.
26. Luo, G., Design and evaluation of the iMed intelligent medical search engine. *Proceedings of ICDE'09*, pp. 1379–1390, 2009.
27. Lewis, D., Eysenbach, G., and Kukafka, R., *et al*., *Consumer Health Informatics: Informing Consumers and Improving Health Care*. Springer, 2005.
28. Luo, G., and Tang, C., Automatic home nursing activity recommendation. *Proceedings of AMIA'09*, pp. 401–405, 2009.
29. Luo, G., Tang, C., and Thomas, S. B., Intelligent personal health record: experience and open issues. *JMS*, to appear.
30. Luo, G., Thomas, S. B., and Tang, C., Automatic home medical product recommendation. *JMS*, to appear.
31. WebMD personal health record homepage. http://www.webmd.com/phr, 2011.
32. Microsoft HealthVault homepage. http://www.healthvault.com, 2011.
33. The NLM Personal Health Record (PHR). http://www.lhncbc.nlm.nih.gov/lhc/servlet/Turbine/template/research,medinform,PHR.vm, 2011.
34. Page, L., Brin, S., and Motwani, R., *et al*., The PageRank citation ranking: bringing order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.
35. Ponte, J. M., and Croft, B. W., A language modeling approach to information retrieval. *Proceedings of SIGIR'98*, pp. 275–281, 1998.
36. Paton, N. W., and Díaz, O., Active database systems. *ACM Comput. Surv.* 31(1):63–103, 1999.
37. Portet, F., Reiter, E., Gatt, A., et al., Automatic generation of textual summaries from neonatal intensive care data. *Artif. Intell.* 173(7–8):789–816, 2009.
38. Palmer, J. G., and Spaeder, J. A., Outpatient management of chronic diseases using the telewatch patient monitoring system. *Johns Hopkins APL Technical Digest* 25(3):253–260, 2004.
39. Publish/subscribe. http://en.wikipedia.org/wiki/Publish/subscribe, 2011.
40. RelayHealth personal health record homepage. https://www.patientally.com/Default.aspx?CID=pa, 2011.
41. Reiter, E., and Dale, R., *Building Natural Language Generation Systems*. Cambridge University Press, 2000.
42. Rosso, R., Munaro, G., and Salvetti, O., *et al*., CHRONIOUS: an open, ubiquitous and adaptive chronic disease management platform for chronic obstructive pulmonary disease (COPD), chronic kidney disease (CKD) and renal insufficiency. *Proceedings of EMBC'10*, pp. 6850–6853, 2010.
43. Roy, P., and Sudarshan, S., Multi-query optimization. *Encyclopedia of Database Systems*, pp. 1849–1852, Springer, 2009.
44. Sachs, G. S., and Guille, C., Weight gain associated with use of psychotropic medications. *J. Clin. Psychiatry* 60(Suppl 21):16–19, 1999.
45. Sornmo, L., and Laguna, P., *Bioelectrical signal processing in cardiac and neurological applications*. Academic Press, 2005.
46. Tatro, D. S., *2011 drug interaction facts: the authority on drug interactions*. Lippincott Williams & Wilkins, 2010.
47. United Nations, World population ageing 2009. http://www.un.org/esa/population/publications/WPA2009/WPA2009_WorkingPaper.pdf, 2009.
48. Vogeli, C., Shields, A. E., Lee, T. A., et al., Multiple chronic conditions: prevalence, health consequences, and implications

for quality, care management, and costs. *JGIM* 22(Suppl 3):391–395, 2007.
49. Veen, J. C., Smits, H. H., Ravensberg, A. J., et al., Impaired perception of dyspnea in patients with severe asthma. Relation to sputum eosinophils. *Am. J. Respir. Crit. Care Med.* 158(4):1134–1141, 1998.
50. The World Health Organization, The global shortage of health workers and its impact. http://www.allcountries.org/health/the_global_shortage_of_health_workers_and_its_impact.html, 2006.
51. Bing developer center. http://www.bing.com/developers/, 2011.
52. Turpin, A., Tsegay, Y., and Hawking, D., *et al.*, Fast generation of result snippets in web search. *Proceedings of SIGIR'07*, pp. 127–134, 2007.
53. International Classification of Diseases (ICD-10) homepage. http://www.who.int/classifications/icd/en/, 2011.
54. Luo, G., Tang, C., and Tian, Y., Answering relationship queries on the web. *Proceedings of WWW'07*, pp. 561–570, 2007.
55. Luo, G., On search guide phrase compilation for recommending home medical products. *Proceedings of EMBC'10*, pp. 2167–2171, 2010.