Data In, Fact Out:
Automated Monitoring of Facts by FactWatcher

Naeemul Hassan [1], Afroza Sultana [1], You Wu [2], Gensheng Zhang [1]
Chengkai Li [1], Jun Yang [2], Cong Yu [3]

[1]*University of Texas at Arlington,* [2]*Duke University,* [3]*Google Research*

ABSTRACT

Towards computational journalism, we present FactWatcher, a system that helps journalists identify data-backed, attention-seizing facts which serve as leads to news stories. FactWatcher discovers three types of facts, including situational facts, one-of-the-few facts, and prominent streaks, through a unified suite of data model, algorithm framework, and fact ranking measure. Given an append-only database, upon the arrival of a new tuple, FactWatcher monitors if the tuple triggers any new facts. Its algorithms efficiently search for facts without exhaustively testing all possible ones. Furthermore, FactWatcher provides multiple features in striving for an end-to-end system, including fact ranking, fact-to-statement translation and keyword-based fact search.

1. MOTIVATION

Computational journalism [1, 2] is a young field that assists journalism using computing. One of its objectives is to find news leads backed up by hard, factual data. In the last several years, our research in this thrust has been focused on automatic and algorithmic fact finding by database and data mining techniques [3, 5, 4, 6]. Specifically, we studied how to monitor three types of facts that can be expressed as the following factual statements:

Situational fact [4] "The social world's most viral photo ever generated 3.5 million likes, 170,000 comments and 460,000 shares by Wednesday afternoon." (http://www.cnbc.com/id/49728455) A situational fact is about a *contextual skyline* object within a certain context (e.g., all photos posted to Facebook) with regard to several measures (e.g., number of "likes", number of "comments", and number of "shares"), i.e., the object is not *dominated* by any object in the context when they are compared by the measures.

One-of-the-few [5] "Victor Oladipo scored 30 points and handed out 14 assists ... only three other rookies have recorded at least 30 points and 14 assists in a game ..." (http://espn.go.com/espn/elias?date=20140222) This statement is about a one-of-the-four object, which is only dominated by at most three other objects.

Prominent streak [3, 6] "This month the Chinese capital has experienced 10 days with a maximum temperature in around 35 degrees Celsius—the most for the month of July in a decade." (http://www.chinadaily.com.cn/china/2010-07/27/content_11055675.htm) A prominent streak is a long consecutive subsequence (e.g., 10 days of temperature) consisting of only large (small) values (e.g., all above a value close to 35 degrees) within a sequence of values (e.g., daily maximum temperature of Beijing).

Proceedings of the VLDB Endowment, Vol. 7, No. 13

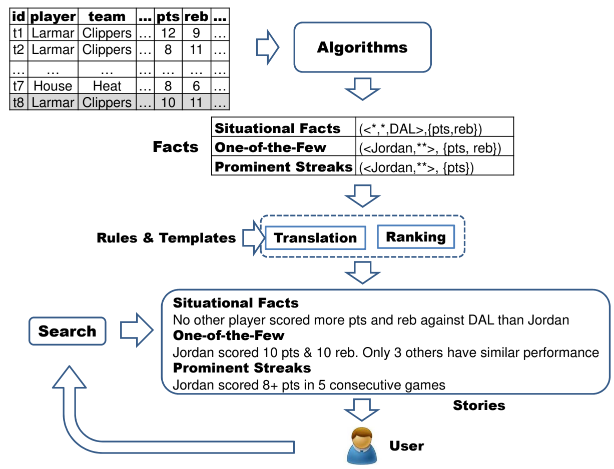

Figure 1: FactWatcher **System Architecture**

Automatic fact finding is helpful in multiple news domains, as factual statements similar to the above ones can be found from not only social media, sports and weather data, but also criminal records, government records and stock data. Several more examples are 1) situational fact: "There were 35 DUI arrests and 20 collisions in city C yesterday, the first time in 2013." 2) one-of-the-few: "Rick Perry is one of the only three candidates in the 2012 US federal election cycle to have received at least $600k from 'lawyers & lobbyists' (an interest group that is usually pro-Democrat) and $400k from 'energy & natural resources' (usually pro-Republican)." and 3) prominent streak: "The Nikkei 225 closed below 10000 for the 12th consecutive week, the longest such streak since June 2009."

This paper demonstrates FactWatcher, a system for automated monitoring of the aforementioned three types of facts. Figure 1 illustrates the components of FactWatcher. Given an ever-growing database, upon the arrival of a new tuple t, FactWatcher checks if t triggers any new situational facts, one-of-the-few facts, and prominent streaks. It is impossible (especially with a manual approach) to exhaustively check all possible facts upon the arrival of every new tuple in a large database, due to the large search space. The systematic and efficient algorithms in FactWatcher thus enable a practical tool that assists journalists in identifying newsworthy stories. This is particularly valuable when we consider the diminishing readership and resources that traditional news media is facing.

FactWatcher is an integrated system beyond the piecemeal algorithms in [3, 5, 4, 6]. It incorporates all three types of facts

id	player	team	opposition team	pts	ast	reb
t_1	*Lamar Odom*	*Clippers*	*Nets*	12	9	13
t_2	*Lamar Odom*	*Clippers*	*Lakers*	8	11	6
t_3	*Lamar Odom*	*Clippers*	*Lakers*	9	9	7
t_4	*Eddie House*	*Heat*	*Nets*	9	7	8
t_5	*Lamar Odom*	*Heat*	*Nets*	10	11	12
t_6	*Eddie House*	*Clippers*	*Nets*	10	11	10
t_7	*Eddie House*	*Heat*	*Wizards*	8	6	9
t_8	*Lamar Odom*	*Clippers*	*Lakers*	10	11	11

Table 1: A Data Table for the Running Example

into a unified suite of data model, algorithm framework and fact ranking measure. It enables monitoring one-of-the-few facts in all different subspaces of dimension and measure attributes, which was not considered in [5]. It also supports a novel measure for ranking all types of facts by the elapsed time since their last comparable facts were discovered. Furthermore, FactWatcher provides multiple features in striving for an end-to-end system. By using rules and templates, the discovered facts are translated into textual news leads and presented to users; it allows users to customize the system by choosing which attributes in the database to consider and which measures to employ in ranking facts; it also supports keyword-based search of facts.

2. CONCEPTS

Consider an append-only table $R(\mathcal{D};\mathcal{M})$, where $\mathcal{D}$=$\{d_1,\ldots,d_n\}$ is a set of *dimension attributes* and $\mathcal{M} = \{m_1,\ldots,m_q\}$ is a set of *measure attributes*. A *constraint* C is a conjunctive expression of the form $d_1{=}v_1\wedge\ldots\wedge d_n{=}v_n$ (also written as $\langle v_1,v_2,\ldots,v_n\rangle$ for simplicity), where $v_i{\in}dom(d_i){\cup}\{*\}$ and $dom(d_i)$ is the value domain of dimension attribute d_i. The set of all possible constraints is denoted $\mathcal{C}$. Given a constraint $C\in\mathcal{C}$, $\sigma_C(R)$ is the relational algebra expression that chooses all tuples in R that satisfy C.

Given a *measure subspace* $M{\subseteq}\mathcal{M}$ and two tuples $t,t'{\in}R$, t *dominates* t' with respect to M, denoted by $t{\succ_M}t'$ or $t'{\prec_M}t$, if t is not worse than t' on any measure attribute in M and t is better than t' on at least one measure attribute. Let $\delta_M(R,t)$ denote the number of tuples in R that dominate t with regard to M. The *k-skyband* ($k{\geq}1$) of R in M, denoted $\lambda^k_M(R)$, is the set of tuples in R dominated by fewer than k other tuples, i.e., $\lambda^k_M(R){=}\{t{\in}R \mid \delta_M(R,t){<}k\}$. The 1-skyband ($\lambda^1_M(R)$, or simply $\lambda_M(R)$) is known as the *skyline* of R. Given a user-specified threshold $\tau\geq 1$, the *top-τ skyband* of R in M, denoted $\tau_M(R)$, is the $\hat{k}$-skyband where $\hat{k}{=}\max\{k \mid \tau \geq |\lambda^k_M(R)|\}$, i.e., $\hat{k}$ is the largest such integer that the $\hat{k}$-skyband has no more than τ tuples. (For rigor, we say $\lambda^0_M(R){=}0$.)

When a new tuple t is appended to R, FactWatcher discovers three types of interesting facts about t, as follows.

Situational fact FactWatcher finds $S^t{=}\{(C,M)\mid C{\in}\mathcal{C},\ M \subseteq\mathcal{M}, t{\in}\lambda_M(\sigma_C(R))\}$—the set of constraint-measure pairs (C,M) such that t is in the corresponding *contextual skyline*, i.e., the skyline of those objects satisfying C with regard to M. Consider Table 1, where $\mathcal{D}$=$\{$*player, team, opposition team*$\}$ and $\mathcal{M}$=$\{$*pts, ast, reb*$\}$. The last appended tuple t_8 belongs to the contextual skylines for several constraint-measure pairs, including ($\langle$**, *, Lakers*$\rangle$, $\{$*pts, ast*$\}$) and ($\langle$*Lamar Odom, Clippers, **$\rangle$, $\{$*ast*$\}$).

One-of-the-few FactWatcher discovers $F^t{=}\{(C,M)\mid C\in\mathcal{C}, M{\subseteq}\mathcal{M}, t{\in}\tau_M(\sigma_C(R))\}$—the set of such constraint-measure pairs (C,M) that t is in the corresponding top-τ skyband. Consider Table 1 again. For constraint-measure pair ($\langle$*Lamar Odom, *, **$\rangle$, $\{$*pts, reb*$\}$), the skyline, 2-skyband and 3-skyband are $\{t_1\}$, $\{t_1,t_5\}$ and $\{t_1,t_5,t_8\}$, respectively. Hence, t_8 belongs to the top-3 skyband but not the top-2 skyband of this constraint-measure pair.

Prominent streak Given a set of object identification attributes $I{\subseteq}\mathcal{D}$, $\mathcal{G}{=}\{I\cup S \mid S{\in}2^{\mathcal{D}-I}\}$ defines all considered ways of grouping tuples. In Table 1, if I=$\{$*player*$\}$, then $\mathcal{G}$=$\{\{$*player*$\}$, $\{$*player, team*$\}$, $\{$*player, opposition team*$\}$, $\{$*player, team, opposition team*$\}\}$. Given grouping attributes $G{\in}\mathcal{G}$, the corresponding group-by and aggregation operation is denoted $seq\ \gamma_G(R)$. seq is an aggregate function that, for each distinct value g of G (i.e., a group), produces an *ordered sequence* $P_g{=}t_{e_1}\ldots t_{e_u}$ consisting of all tuples in the group, i.e., $\forall 1{\leq}i{\leq}u,\ t_{e_i}[G]{=}g$. The tuples are ordered by their unique timestamps—$\forall 1{\leq}i{<}j{\leq}u$, t_{e_i} was inserted into R before t_{e_j}, i.e., the real-world event for t_{e_i} happened before that for t_{e_j}.

A *streak* s in a sequence $P_g{=}t_{e_1}\ldots t_{e_u}$ is any consecutive subsequence $t_{e_l}\ldots t_{e_r}$. We use $s.len$ to denote the length of s (i.e., $r{-}l{+}1$). We use $s.\vec{v}$ to denote the vector containing the minimal value in s on every measure attribute, i.e., $s.\vec{v}{=}(\min_{l\leq i\leq r}t_{e_i}[m_1]$, $\ldots$, $\min_{l\leq i\leq r}\ t_{e_i}[m_q])$, where $\{m_1,...,m_q\}$ forms the set of all measure attributes $\mathcal{M}$. Given a set of streaks S and a measure subspace $M{\subseteq}\mathcal{M}$, $s{\in}S$ is *prominent* if s is not dominated by any other streak (i.e., s is a skyline streak in S), where the *dominance* relation is based on streaks' lengths and the projections of their minimal value vectors on M. Hence, we use $\lambda_{\{len\}\cup M}(S)$ to denote the set of all prominent streaks in S with regard to M.

Upon arrival of the latest tuple t, FactWatcher discovers $P^t{=}\{(G,M)\mid G{\in}\mathcal{G}, M{\subseteq}\mathcal{M}, \exists\, s{\in}\lambda_{\{len\}\cup M}(S(P_{t[G]}))$ s.t. $s{=}\ldots t\}$, where $S(P_{t[G]})$ denotes the set of all streaks in sequence $P_{t[G]}$. In other words, with regard to G and M, the arrival of t establishes one or more new prominent streaks that end at t. Suppose the tuples in Table 1 are ordered by their ids. Consider I=$\{$*player*$\}$, G=$\{$*player, team*$\}$ and group g for *player*=*Lamar Odom*, *team* = *Clippers*. Before arrival of t_8, $P_g{=}t_1t_2t_3$ and thus $S(P_g){=}\{t_1,t_2,t_3,\ t_1t_2,t_2t_3,t_1t_2t_3\}$. The prominent streaks with regard to M=$\{pts, ast\}$ are $\lambda_{\{len,pts,ast\}}$ $(S(P_g)){=}\{t_1,t_2,t_1t_2t_3\}$. Note that steak $s{=}t_1t_2$ ($s.len{=}2$, $s.\vec{v}{=}(8,9)$) is not prominent because it is dominated by streak $s'{=}t_1t_2t_3$ ($s'.len{=}3$, $s'.\vec{v}{=}(8,9)$). Upon arrival of t_8, $P_g = t_1t_2t_3t_8$, and the prominent streaks $\lambda_{\{len,pts,ast\}}$ $(S(P_g))$ become $\{t_1,t_8,t_1t_2t_3t_8\}$. There are more than one new prominent streaks ending at t_8, corresponding to facts related to t_8.

3. USER INTERFACE

Figure 2 shows FactWatcher's customized GUI for NBA (National Basketball Association) data, where new tuples—players' statistics in individual games—come into the database when a game is over. The GUI's structure is dataset-agnostic as long as the data table is modeled by $R(\mathcal{D};\mathcal{M})$ as given in Section 2. For instance, for data analytics of a social network service, suppose the dimension attributes are $\mathcal{D}$=$\{$*user age, user city, post type, timestamp*$\}$ and the measure attributes are $\mathcal{M}$=$\{$*number of likes, number of comments, number of shares*$\}$. FactWatcher finds facts such as "no one in Dallas has posted a photo that gets as many likes, comments and shares." The GUI consists of four areas, as follows.

1. Stories This area shows a ranked list of textual news leads (stories) translated from facts that have ever been discovered and are still valid. It also allows users search for translated stories by keywords. The translation is guided by a set of templates and rules. For example, if t_8 in Table 1 triggers a situational fact with regard to constraint-measure pair ($\langle$**, Clippers, Lakers*$\rangle$, $\{$*pts, ast, reb*$\}$), the story is "Lamar Odom had 10 points, 11 assists and 11 rebounds to become the first Clippers player with a 10/11/11 (points/ assists/ rebounds) game against the Lakers."

If a story is clicked, FactWatcher shows below it more stories for the same constraint-measure pair (C,M) or grouping-measure pair (G,M), as illustrated in Figure 2. It also presents bar charts to compare the stories by their values on M.

2. Ranking FactWatcher allows users to choose from several ways of ranking facts and their corresponding stories.

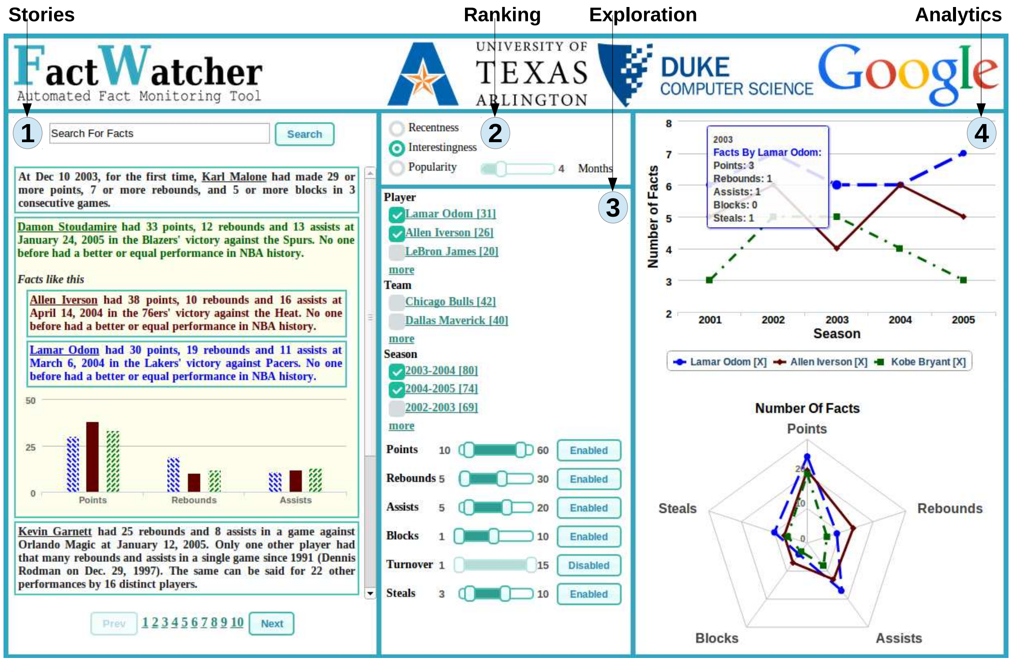

Figure 2: FactWatcher **User Interface**

Recentness This default option simply orders facts by their triggering tuples' timestamps.

Popularity This option ranks facts by the frequencies of facts appearing in search results within the last x months, where x can be controlled using a slidebar.

Interestingness This option ranks facts by the elapsed time since their last comparable facts were discovered. Suppose tuple t triggers a new situational fact f_1 with regard to constraint-measure pair (C, M) and a new prominent streak f_2 with regard to grouping-measure pair (G, M). The interestingness of f_1 (f_2) is the elapsed time since the last fact was discovered in (C, M) ((G, M)). The longer the elapsed time is, the more interesting a fact is, since a long elapsed time indicates the fact does not come by easily.

3. Exploration This area presents a faceted interface for exploring the stories. Each facet corresponds to a dimension or measure attribute. Under the facet for a dimension attribute, the attribute values are associated with and ordered by numbers, which indicate how many facts involve the values. For instance, Figure 2 shows that there are 31 facts for such (C, M) that the constraint C has a conjunct *player*=*Lamar Odom*. FactWatcher places a checkbox beside each attribute value. A user can select/unselect the checkboxes across multiple facets. The selected values within one facet correspond to a disjunctive condition, and the disjuncts from different facets form a conjunctive condition. They together correspond to multiple constraints. Each fact (story) displayed in the "stories" area must satisfy one such constraint.

Beside the facet for a measure attribute, FactWatcher presents a slidebar and a button. A user can click the button to enable or disable a measure attribute. Accordingly the "stories" area displays such stories whose corresponding measure subspaces M only involve one or more enabled measure attributes. The user can also use the slidebar to set the minimum and maximum desired values on an enabled attribute m_i. Accordingly the displayed facts (stories) must have values on m_i within the range.

4. Analytics This area visualizes statistics on facts related to objects selected by a user. The "stories" area highlights the objects (values on object identification attributes) in stories, e.g., *Allen Iverson, Lamar Odom*, etc. in Figure 2. When a user clicks on an object, it is added to the object list in the middle of the "analytics" area. The user can remove an object by clicking the crossmark beside it. The top part of the "analytics" area is a line chart, which shows one line per selected object that represents the number of facts (among the displayed facts in the "stories" area) triggered by the object over each time period. When the user hovers the mouse on a data point, a pop-up box shows, for each measure attribute, the number of facts whose measure subspaces contain the measure attribute. The bottom part of this area is a radar chart, which shows one polygon per selected object that represents how many facts triggered by the object are related to each measure attribute.

4. ALGORITHMS

Situational fact

In finding situational facts upon the arrival of a new tuple t, a brute-force approach would compare t with all existing tuples to determine whether t is dominated, repeatedly for each constraint C satisfied by t in each measure subspace M. This approach is clearly exhaustive due to comparison with every tuple, for every constraint, and in every measure subspace. The algorithms in FactWatcher respond to these inefficiencies by the following ideas.

Tuple reduction Instead of comparing t with every previous tuple, it is sufficient to only compare with current skyline tuples. This is based on the simple property that, if any tuple dominates t, there must exist a skyline tuple that also dominates t. For example, in Table 1, if the context is the whole table (i.e., no constraint) and the measure subspace M={*pts, reb*}, the contextual skyline has one tuple—t_1. When the new tuple t_8 comes, with regard to the same constraint-measure pair, it suffices to compare t_8 with t_1 only.

Constraint pruning If new tuple t is dominated by another tuple t' in a measure subspace M, then t does not belong to the contextual skyline of (C, M) for any constraint C satisfied by both t and t'. For example, since t_8 is dominated by t_1 in the full measure space $\mathcal{M}$, it is not in the contextual skylines of (⟨*Lamar Odom, Clippers*, *⟩, $\{\mathcal{M}\}$), (⟨*Lamar Odom*, *, *⟩, $\{\mathcal{M}\}$), (⟨*, *Clippers*, *⟩, $\{\mathcal{M}\}$), and (⟨*, *, *⟩, $\{\mathcal{M}\}$). Based on this, FactWatcher examines the constraints satisfied by t in a certain order, and comparisons of t with skyline tuples associated with already examined constraints are used to prune remaining constraints.

Sharing computation across measure subspaces FactWatcher considers the full measure space $\mathcal{M}$ first and prunes various constraints for measure subspaces. For instance, after comparing t_8 with t_6 in $\mathcal{M}$, it realizes that t_8 has equal value on *pts* and smaller value on *ast* and thus t_8 is dominated by t_6 in {*pts*, *ast*} and {*ast*} under the constraints satisfied by both tuples.

Based on these ideas, the algorithms in FactWatcher efficiently maintain contextual skylines for all constraint-measure pairs. Upon the arrival of a new tuple t, for all measure subspaces starting from $\mathcal{M}$, constraints satisfied by t (which form a lattice based on subsumption relation between constraints and their corresponding contexts) are visited in either a bottom-up or a top-down order. The new tuple is compared with current skyline tuples associated with the constraints. Various constraints are pruned based on above ideas. Skylines for all constraint-measure pairs are maintained to include t and/or purge current skyline tuples if necessary.

One-of-the-few

While situational facts are about skyline objects, one-of-the-few facts are about top-τ skyband objects. For each constraint-measure pair (C, M), the algorithms maintain the $\hat{k}$-skyband $\lambda_M^{\hat{k}}(\sigma_C(R))$ for such a dynamic $\hat{k}$ that $\lambda_M^{\hat{k}}(\sigma_C(R))$ equals the top-τ skyband $\tau_M(\sigma_C(R))$, i.e., $\hat{k}=\max\{k \mid \tau \geq |\lambda_M^{k}(\sigma_C(R))|\}$. A dominated tuple cannot be immediately discarded. Instead, a counter should be maintained to record $\delta_M(\sigma_C(R), t)$ for tuple t. Below is the core idea of maintaining top-τ skyband for a particular (C, M).

Let R' denote the relation after inserting a new tuple t' into relation R. Suppose t' satisfies constraint C. For any $t \in \sigma_C(R)$, t' may increase $\delta_M(\sigma_C(R), t)$ by at most 1. Hence, if $\tau_M(\sigma_C(R))$ is equal to $\lambda_M^{\hat{k}}(\sigma_C(R))$, $\tau_M(\sigma_C(R'))$ must be (i) $\lambda_M^{\hat{k}}(\sigma_C(R'))$, (ii) $\lambda_M^{\hat{k}-1}(\sigma_C(R'))$, or (iii) $\lambda_M^{\hat{k}+1}(\sigma_C(R'))$. To support incremental computation, we maintain $\lambda_M^{\hat{k}+1}(\sigma_C(R))$ (instead of $\lambda_M^{\hat{k}}(\sigma_C(R))$) and compute $\lambda_M^{\hat{k}+1}(\sigma_C(R'))$ from $\lambda_M^{\hat{k}+1}(\sigma_C(R))$. There are two cases, depending on how many tuples in $\sigma_C(R)$ dominate t':

I. $\delta_M(\sigma_C(R), t') \geq \hat{k}$. By definition, $t' \notin \tau_M(\sigma_C(R'))$, as it cannot dominate any tuple $t \in \tau_M(\sigma_C(R))$. In this case, $\tau_M(\sigma_C(R'))= \lambda_M^{\hat{k}}(\sigma_C(R'))=\lambda_M^{\hat{k}}(\sigma_C(R))$.

II. $\delta_M(\sigma_C(R), t') < \hat{k}$. In this case, update $\delta_M(\sigma_C(R'), t)$ for $t \in \lambda_M^{\hat{k}+1}(\sigma_C(R))$, and check if $\hat{k}'$ should be $\hat{k}$, $\hat{k}-1$, or $\hat{k}+1$ for $\tau_M(\sigma_C(R'))$.

Prominent streak

Upon a new tuple t, FactWatcher discovers new prominent streaks for all grouping-measure pairs (G, M). To share computation across different M, a data-cube style data structure and exploration space is adopted. Below we outline the key ideas of how to incrementally monitor prominent streaks for a particular (G, M).

Our solution hinges upon the idea to separate two steps—*candidate streak generation* which generates a small number of candidate streaks ending at the new tuple without exhaustively considering all possible streaks, and *skyline operation* which maintains a dynamic set of prominent streaks by performing dominance comparison between existing prominent streaks and candidate streaks. The effectiveness of pruning in the first step is critical to overall performance, because execution time of skyline algorithms increases superlinearly by number of candidates.

A brute-force approach to candidate streak generation would enumerate all $\frac{n(n+1)}{2}$ possible streaks in an n-tuple sequence as candidates. We proposed the concept of *local prominent streak* (LPS) for substantially reducing candidate streaks. The intuition is, given a prominent streak s, there cannot be a super-sequence of s whose minimal value vector dominates $s.\vec{v}$. In other words, s must be locally prominent as well. Hence we only need to consider LPSs as candidates, the number of which is at most n—the length of the sequence. The algorithm incrementally maintains possible LPSs while new tuples keep getting appended to the database.

5. DEMONSTRATION PLAN

We will demonstrate FactWatcher on several datasets, including an NBA dataset and a weather dataset. The NBA dataset has 317,371 tuples of NBA box scores from 1991-2004, on 8 dimension and 7 measure attributes. The weather dataset has 7.8 million daily weather forecast records for 5,365 locations of UK from Dec. 2011 to Nov. 2012. It has 7 dimension attributes and 7 measure attributes. When we explain the demonstration steps below, we refer to the GUI in Figure 2 and its corresponding NBA scenario.

Stories When a user visits FactWatcher, FactWatcher shows a list of stories in area "stories" of Figure 2. The user enters a keyword query in the search box. The list of stories will be updated. The faceted interface in area "exploration" and the line chart and radar chart in area "analytics" will change accordingly. The user then clicks a particular story. Similar stories will be shown below it, with bar charts to compare the stories.

Ranking By default, the stories are ordered by recentness. The user explores other ranking schemes by choosing the radio button for *interestingness* or *popularity* in area "ranking". When *popularity* is chosen, the user further uses the slidebar beside it to control the period for assessing popularity of stories.

Exploration The user uses the faceted interface in area "exploration" to explore stories. The user checks *Lamar Odom*, *Allen Iverson* and some others under *player* and *2003-2004*, *2004-2005* under *season*. The area "stories" will show stories related to any of the selected players when they played for or against any team (as she did not select anything under *team*) during *2003-2004* or *2004-2005* season. The user further uses the slidebars for measure attributes to adjust the ranges of values on these attributes. The area "stories" will only show those stories whose measure attribute values do not fall out of the ranges. If the user wants to exclude a measure attribute from the filtering criteria, she can click the button beside its slidebar to disable it, e.g., *Turnover* in Figure 2.

Analytics When the user reads the stories, she can click on any underlined objects, i.e., players. After a while, the user has clicked on multiple objects, which are shown in the box in the middle of area "analytics". The top line chart and bottom radar chart in that area visualize the statistics on facts related to these objects, as described in Section 3. If the user is not interested in an object anymore, she can remove the object from comparison by clicking the "X" beside the object in the middle box.

Acknowledgments The authors are supported by NSF grants IIS-09-16027, IIS-10-18865, CCF-11-17369, and IIS-13-20357. Addition support comes from HP Labs Innovation Research Awards, a Google Faculty Research Award, and National Natural Science Foundation of China Grant 61370019. Any opinions, findings, and conclusions in this publication are those of the authors and do not necessarily reflect the views of the funding agencies.

6. REFERENCES

[1] S. Cohen, J. T. Hamilton, and F. Turner. Computational journalism. *Commun. ACM*, 54(10):66–71, Oct. 2011.

[2] S. Cohen, C. Li, J. Yang, and C. Yu. Computational journalism: A call to arms to database researchers. In *CIDR*, pages 148–151, 2011.

[3] X. Jiang, C. Li, P. Luo, M. Wang, and Y. Yu. Prominent streak discovery in sequence data. In *KDD*, pages 1280–1288, 2011.

[4] A. Sultana, N. Hassan, C. Li, J. Yang, and C. Yu. Incremental discovery of prominent situational facts. In *ICDE*, 2014.

[5] Y. Wu, P. K. Agarwal, C. Li, J. Yang, and C. Yu. On one of the few objects. In *KDD*, pages 1487–1495, 2012.

[6] G. Zhang, X. Jiang, P. Luo, M. Wang, and C. Li. Discovering general prominent streaks in sequence data. *ACM TKDD*, to appear.