Effiziente Filterung in zentralisierten und verteilten Benachrichtigungssystemen

Sven Bittner
bittner@inf.fu-berlin.de
Freie Universität Berlin
Fachbereich Mathematik und Informatik

Betreuer der Arbeit: Prof. Dr.-Ing. Heinz F. Schweppe und Dr. Annika Hinze
Art der Arbeit: Diplomarbeit
GI-Fachbereich: Datenbanken und Informationssysteme [DBIS]

Zusammenfassung

Ein großes Problem beim Entwurf eines Benachrichtigungsdienstes ist eine effiziente Filterkomponente. Insbesondere bei Realzeitsystemen, zum Beispiel der Gebäudesteuerung, ist die Effizienz des eingesetzten Benachrichtigungsdienstes eines der wichtigsten Entwurfskriterien. Im Rahmen dieser Arbeit wird ein Benachrichtigungssystem entwickelt. Dabei wird besonders die effiziente Filterung, zuerst in einem zentralisierten System und danach in einer verteilten Umgebung betrachtet. In der Literatur vorhandene Algorithmen werden dazu erweitert bzw. neue Lösungsideen vorgeschlagen. Als Ergebnis entsteht der Prototyp DAS eines verteilten Benachrichtigungssystems.

1. Einführung

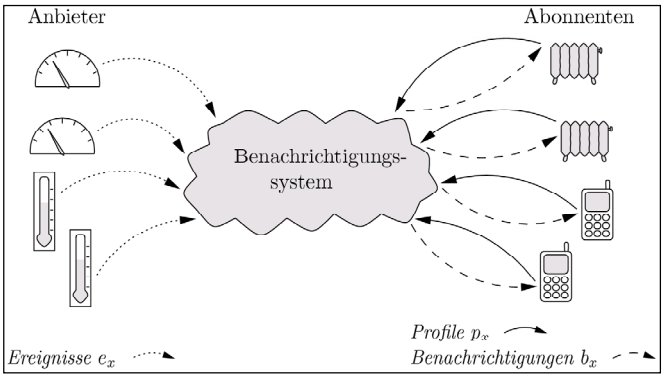

Abb. 1: Übersicht eines Benachrichtigungssystems im Kontext der Gebäudesteuerung.

In den letzten Jahren erhalten *Benachrichtigungsdienste* immer mehr Aufmerksamkeit, z.B. in digitalen Bibliotheken, der Gebäudesteuerung oder der Verkehrskontrolle. Abb. 1 zeigt ein solches Benachrichtigungssystem im Kontext der Gebäudesteuerung. Diese Systeme informieren ihre *Abonnenten* beim Auftreten von bestimmten *Ereignissen*, in der Gebäudesteuerung beispielsweise einer Temperaturänderung. Die Ereignisse werden dem System von *Anbietern*, beispielsweise Temperatursensoren in einem Arbeitszimmer, zur Verfügung gestellt. Die Darstellung der Ereignisse erfolgt durch *Attribut-Wert-Paare*. Dabei ist jedem Attribut *a* ein Wertebereich *W(a)* zugeordnet. Die Ereignisstruktur beschreibt der eindeutige *Ereignistyp*. Die Menge aller möglichen Ereignisse wird mit *E* bezeichnet.

Die Abonnenten eines Benachrichtigungssystems sind an einer Teilmenge der Ereignisse interessiert. Ihre Interessen bestimmen sie in Form von *Profilen*, welche am Benachrichtigungssystem angemeldet werden. Diese werden durch einen Ereignistyp und eine Menge an *Prädikaten* für die Attribute dieses Typs beschrieben. In dieser Arbeit werden in Prädikaten die Operatoren Gleichheit, Größer, Kleiner, der Bereichstest und der Mengentest unterstützt. Werten alle Prädikate eines Profils p_x auf einem Ereignis e_x mit *Wahr* aus, so erfüllt Ereignis e_x Profil p_x ($e_x \succ p_x$) und der Abonnent von p_x wird benachrichtigt. Man sagt auch Profil p_x passt zu Ereignis e_x. Das Finden aller passenden Profile zu den Ereignissen wird als *Filterung* bezeichnet. Die Auslieferung der Ereignisse an die jeweiligen Abonnenten als *Benachrichtigung* b_x. Die Effizienz der Filterung stellt ein wichtiges Gütekriterium eines Benachrichtigungssystems dar und ist entscheidend für dessen Einsatzmöglichkeiten in Realzeitsystemen, wie der Gebäudesteuerung.

Ein *verteilter Benachrichtigungsdienst* ist ein Netz von mehreren Servern, auch Vermittler *V* genannt. Die Vermittler sind über eine beliebige Netztopologie miteinander verbunden. Im Rahmen dieser Arbeit werden als Topologie azyklische zusammenhängende Graphen betrachtet. Abonnenten und Anbieter kommunizieren in einem verteilten Benachrichtigungsdienst mit einem Vermittler ihrer Wahl (lokaler Vermittler). Mit diesem verbundene Anbieter und Abonnenten werden allgemein lokale Klienten genannt.

Im Rahmen dieser Arbeit [Bit03] wird ein neuer zentralisierter Filteralgorithmus vorgeschlagen. Weiterhin werden neue Verfahren zur Verminderung von Profilredundanzen in einem verteilten Benachrichtigungssystem vorgestellt. In DAS werden drei verteilte Filterstrategien unter Nutzung der entwickelten zentralisierten Filterkomponente realisiert. Diese werden in zahlreichen Experimenten unter der Variation verschiedener Systemparameter getestet und bewertet. Dadurch werden Lücken in vorhandenen Arbeiten geschlossen, welche stets nur unzureichende Evaluationen von Filteralgorithmen durchführen.

2. Zentralisierte Filterung

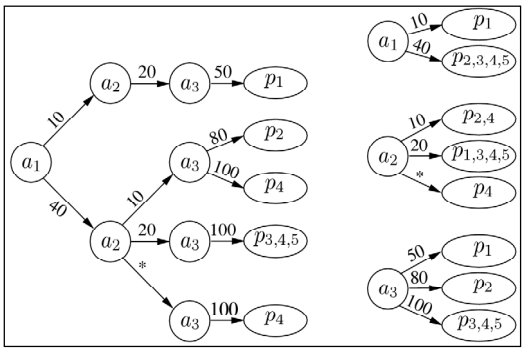

Abb. 2: Filterbaum nach Gough und Smith (links) und Erweiterung (rechts) mit den Profilen p_1 : (a_1=10∧a_2=20∧a_3=50,T_1), p_2 : (a_1=40∧a_2=10∧a_3=80,T_1), p_3 : (a_1=40∧a_2=20∧a_3=100,T_1), p_4 : (a_1=40∧a_3=100,T_1), p_5 : (a_1=40∧a_2=20∧a_3=100,T_1).

Die schnellsten derzeit bekannten zentralisierten Filteralgorithmen nutzen baumbasierte Filterstrukturen [GS95]. Dazu werden die Prädikate der Profile attributweise in einen ereignistypabhängigen Baum eingetragen. Jede Baumebene ist für die Filterung eines Attributes zuständig. Abb. 2 zeigt links einen Filterbaum für fünf Profile mit drei Attributen a_1, a_2 und a_3 des Typs T_1. Ebene 1 des Baumes filtert Attribut a_1, Ebene 2 das Attribut a_2 und Ebene 3 Attribut a_3. Die 4. Ebene bilden die Blätter des Baumes, in denen die Profile eingetragen werden. Zum Filtern von Ereignissen wird, ausgehend von der Baumwurzel, die Kante des aktuellen Knotens verfolgt, die zum Attributwert des Ereignisses passt. Danach wird das Attribut des Folgeknotens auf dem Ereignis ausgewertet. Erreicht man ein Blatt, sind in diesem die passenden Profile eingetragen. Anderenfalls existieren keine passenden Profile. Da Profile nicht alle Attribute spezifizieren müssen, existieren für diesen Fall *-Kanten im Baum. Diese werden beim Filtern verfolgt, falls keiner der Werte der Kanten zum Wert des Attribut-Wert-Paares des Ereignisses passt.

Als eine Erweiterung werden in DAS die Kanten des Baumes nicht mit Attributwerten, sondern mit Bereichen beschrieben. Dadurch können problemlos neben der Gleichheit die o.g. Operatoren unterstützt werden. Die Werte der Kanten sind durch ein Feld *Arr* (der Größe |*Arr*|) von Elementen realisiert. Der Eintrag Arr_i des Index i beschreibt dabei das halboffene Intervall der Werte $]Arr_{i-1}, Arr_i]$ bzw. falls $i = 0$, dann $]-\infty, Arr_i]$. Die Verweise der Kanten werden in einem Feld *Arr'* der gleichen Größe gespeichert. Der Verweis des Index i, Arr_i', ist dabei dem Intervall, beschrieben durch Arr_i, zugeordnet.

Das Problem eines baumbasierten Ansatzes ist sein großer Hauptspeicherbedarf. Deshalb wird ein erweiterter Algorithmus entwickelt, der weniger Hauptspeicher als der von Gough und Smith [GS95] vorgeschlagene Algorithmus benötigt. Im Kontrast zur besseren Speicherausnutzung wird jedoch die Filterzeit erhöht. Dieser Algorithmus baut keine Baumstruktur auf, sondern erzeugt einen Filterknoten für jedes Attribut. Abb. 2 zeigt rechts diese Filterstruktur bei den drei Attributen und fünf Profilen des Typs T_1. Als Vereinfachung sind die Kanten lediglich mit Werten beschrieben, die Erweiterung auf Intervalle ist nicht dargestellt. Um ein Ereignis zu filtern, werden alle Filterknoten der Attribute des Ereignistyps abgearbeitet. Dazu wird der jeweils passenden Kante gefolgt (realisiert durch die Felder *Arr* und *Arr'*). Die dabei erhaltenen Profilmengen in den Blättern werden dann sukzessive geschnitten. Das Resultat ist die Menge der zu allen Attributen passenden Profile. In Abb. 2 erhält man beim Filtern des Ereignisses e_1 : (a_1=40,a_2=10,a_3=80,T_1) folgende Profilmenge: $\{p_2, p_3, p_4, p_5\} \cap \{p_2, p_4\} \cap \{p_2\} = \{p_2\}$.

Eine weitere Verminderung des Speicherbedarfs wird in DAS durch die Verwaltung der Profilverweise in den Blättern der Filterstruktur erreicht. Je nach Anzahl der Verweise werden direkte Zeiger auf Profile (wenige Verweise in einem Blatt) oder Bitlisten (große Verweisanzahl in einem Blatt) genutzt. Weiterhin wird die ursprünglich als statische Komponente konzipierte Filterstruktur zu einer dynamischen Struktur erweitert. Dadurch können Profile problemlos jederzeit hinzugefügt und entfernt werden.

3. Verteilte Filterung

In der Literatur sind folgende verteilte Filteralgorithmen zu finden: Ereignisweiterleitung [CRW99], Profilweiterleitung [CRW99] und Rendezvousknoten [PB02]:

Profilweiterleitung: Bei der Profilweiterleitung (PW) werden Profile im gesamten Vermittlungsnetz verteilt. Tritt ein Ereignis auf, so ist der Vermittlungsknoten des Anbieters für die Filterung verantwortlich. Jeder Vermittler führt die Filteroperationen durch, um Klienten und alle Nachbarn mit lokalen Abonnenten mit passenden Profilen, oder solchen Nachbarn ihrerseits, benachrichtigen zu können. Abonnenten passender Profile werden dann auf dem entgegengesetzten Weg des Pfades, den das Profil zum filternden Vermittlungsknoten zurückgelegt hat, über ein Ereignis benachrichtigt. Die Filterung findet so nah bei den Anbietern wie möglich statt [CRW99].

Ereignisweiterleitung: Die Ereignisweiterleitung (EW) verfolgt die entgegengesetzte Strategie zur Profilweiterleitung. Ereignisse werden im gesamten Netz verbreitet, Profile hingegen von lokalen Vermittlern nicht weitergeleitet. Abonnenten passender Profile werden von ihren lokalen Vermittlern benachrichtigt, eine Weiterleitung von Benachrichtigungen im Vermittlungsnetz erfolgt nicht. Die Filterung findet also so nah bei den Abonnenten wie möglich statt [CRW99].

Rendezvousknoten: Bei Rendezvousknoten (RK) existieren spezialisierte Knoten im Vermittlungsnetz, welche für bestimmte Ereignistypen zuständig sind. Für jeden Typen existiert genau ein Rendezvousknoten im Vermittlungsnetz. Rendezvousknoten fungieren als Treffpunkt für Ereignisse und Profile ihrer Typen. Dazu werden Ereignisse e_x und Profile p_x von lokalen Vermittlern auf dem kürzesten Weg zum Rendezvousknoten ihres jeweiligen Typs gesendet. Die Benachrichtigung über ein Ereignis erfolgt auf dem entgegengesetzten Pfad, auf welchem das zugehörige Profil auf dem Weg zum Rendezvousknoten im Vermittlungsnetz weitergeleitet wurde.

Zur weiteren Verbesserung der Filtereffizienz existieren Optimierungen zur Verminderung von Profilredundanzen in den Vermittlern [MFB02]: Äquivalenz, Bedecken und Gleichheit. Von diesem Optimierungen wird das Bedecken in DAS genutzt: Ein Profil p_x bedeckt [MFB02] ein Profil p_y (geschrieben $p_x \triangleright p_y$), wenn $\{e_x \in E | e_x \succ p_x\} \supseteq \{e_x \in E | e_x \succ p_y\}$. Zur Berechnung der Bedeckungen wird ein neuer Ansatz, die bereichsbasierte Berechnung vorgeschlagen.

Bereichsbasierte Berechnung

Die bereichsbasierte Berechnung setzt auf der entwickelten Filterstruktur auf. Dazu werden die Profilverweise (bzw. die Bitlisten) in den Blättern der Filterstruktur der Vermittler analysiert. Abhängig, welche Verweise ein Profil beinhalten oder nicht, kann damit eine Aussage über die Bedeckung getroffen werden. Zusätzlich muss dazu für jedes Profil die Anzahl der Gesamtverweise in den Blättern gespeichert werden (oder bei jeder Berechnung wird diese Zahl ermittelt). Die Berechnung der von einem Prädikat bedeckten Prädikate (falls dies für alle Prädikate zweier Profile zutrifft bedecken sich dann die Profile), abhängig vom genutzten Vergleichsoperator, geschieht wie folgt. $Arr(w)$ bezeichnet dabei den Index des Elementes w im in Abschnitt 2 vorgestellten Feld mit Kantenbeschriftungen. Aus Platzgründen werden lediglich die Berechnungen beim Gleichheits- und Größeroperator vorgestellt:

Gleichheit (=): Sei das Prädikat $pr = (a, =, w)$ und $i = Arr(w)$. Falls w in Arr enthalten ist ($i \neq -1$) und dessen Intervall Arr_i genau ein Element beschreibt, werden alle Profile aus Arr_i', die in keinem Arr_j' mit $j \neq i$ enthalten sind, bedeckt. Anderenfalls werden keine Profile bedeckt.

Größer (>): Sei das Prädikat $pr = (a, >, w)$ und $k = Arr(\text{succ}(w))$ bzw. des nächstgrößeren Elements. Falls $\text{succ}(w) = \min(W(a))$, werden alle Profile bedeckt. Sonst werden alle Profile bedeckt, deren Zahl der Vorkommen in Arr_j' mit $k <= j < |Arr'|$ gleich der Zahl der Gesamtvorkommen in Arr' ist.

4. Experimente

Zuerst wurde die zentralisierte Filterkomponente zahlreichen Effizienztests unterzogen. Als Vergleichssysteme dienen das populäre Elvin4-System [SAB+00] und die OpenORB Implementierung [Mod00] des CORBA-Notification-Service. Alle drei Systeme zeigen gleichartige Änderungen der Filtereffizienz bei der Variation diverser Systemparameter. Das OpenORB-System zeigt gravierend schlechtere Filtereffizienz als DAS, Elvin4 ist ca. 8-mal langsamer als DAS. Weiterhin wurden die drei verteilten Filteralgorithmen in zahlreichen Experimenten getestet. Es wurden Filtereffizienz, Netzlast beim Veröffentlichen von Ereignissen und Speicherauslastung untersucht. Dabei wurden Variationen zahlreicher Systemparameter durchgeführt: Anteil passender Profile, Anteil passender Ereignisse, Vermittlerzahl, Überdeckungen zwischen Profilen, Anzahl der Ereignistypen, Lokalitätsverhalten zwischen Ereignissen und Profilen und Gesamtprofilanzahl. Abb. 3 zeigt die Filtereffizienz in Abhängigkeit vom Anteil erfüllender Ereignisse und der Vermittlerzahl.

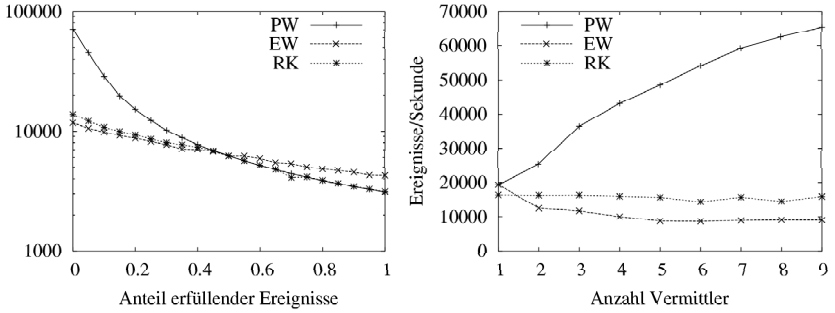

Abb. 3: Filtereffizienz abhängig vom Anteil erfüllender Ereignisse bzw. der Vermittlerzahl.

Erfüllen viele Ereignisse Profile (Abb. 3 links), zeigt die Ereignisweiterleitung die beste Filtereffizienz, sonst die Profilweiterleitung. Ein Effizienzvorteil durch Hinzunahme von Vermittlern ist lediglich bei der Profilweiterleitung (Abb. 3 rechts) zu beobachten. Zusammenfassend wurden folgende Ergebnisse erzielt:

Die Profilweiterleitung zeigte in vielen Untersuchungen beste Filtereffizienz und Netzlast, jedoch großen Speicherbedarf. Die Ereignisweiterleitung zeigte bei hohem Anteil passender Ereignisse bessere Filtereffizienz (einfaches Protokoll) als die anderen Algorithmen und bei hoher Profilanzahl bessere Skalierbarkeit (keine Profilredundanzen). Die Netzlast ist stets sehr hoch, der Speicherbedarf optimal. Die Rendezvousknoten konnten unter keiner der untersuchten Systemkonfigurationen bessere Ergebnisse als andere Verfahren erreichen.

Deshalb sollte ein Benachrichtigungssystem unterschiedliche Filteralgorithmen unterstützen und je nach Systemlast und -nutzung bzw. der aktuellen Anwendung den verwendeten Filteralgorithmus auswählen. Im Anwendungsgebiet der Gebäudeverwaltung erreicht die Profilweiterleitung, aufgrund der dort typischen Profildefinitionen, die besten Ergebnisse.

Literatur

[Bit03] Bittner, S.: Entwurf und Analyse eines effizienten verteilten Benachrichtigungssystems. Diplomarbeit, Freie Universität Berlin, Institut für Informatik, September 2003.

[CRW99] Carzaniga, A.; Rosenblum, D. S.; Wolf, A. L.: Interfaces and Algorithms for a Wide-Area Event Notification Service. Techn. Ber. CU-CS-888-99, Fachbereich Informatik, Universität Colorado, Oktober 1999. Überarbeitet Mai 2000.

[GS95] Gough, J.; Smith, G.: Efficient Recognition of Events in a Distributed System. In: Proceedings of the 18th Australasian Computer Science Conference (ACSC-18), Adelaide, Australien, 1.-3. Februar 1995.

[MFB02] Mühl, G.; Fiege, L.; Buchmann, A.: Filter Similarities in Content-Based Publish/Subscribe Systems. In: Proceedings of the International Conference on Architecture of Computing Systems (ARCS '02), Karlsruhe, Deutschland, 8.-12. April 2002; S. 224-238.

[Mod00] Modica, O.: OpenORB Notification Service Documentation, 8. August 2000. Intalio Inc.

[PB02] Pietzuch, P.; Bacon, J.: Hermes: A Distributed Event-Based Middleware Architecture. In: Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW '02), Wien, Österreich, 2.-5. Juli 2002; S. 611-618.

[SAB+00] Segall, B.;Arnold, D;Boot, J.; Henderson, M; Phelps, T.: Content Based Routing with Elvin4 . In: Proceedings of Australian UNIX and Open Systems User Group Conference (AUUG2K), Canberra, Australien, 25.-30. Juni 2000.