Douglas Cunningham, Petra Hofstedt, Klaus Meer, Ingo Schmitt(Hrsg.): INFORMATIK 2015
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2015

Reservierungsmanager für Cloud-Ressourcen mittels constraintbasierter Programmierung

Hans-Joachim Goltz und Armin Wolf[1]

Abstract: Es wird ein Reservierungsmanager vorgestellt, der Reservierungen von Ressourcen in einer Cloud verwaltet und eine Ressourcenplanung ermöglicht. Die Hauptaufgabe des Reservierungsmanagers besteht darin zu überprüfen, ob für jeden Dienst mit einem vertraglich vereinbarten Fertigstellungstermin die erforderlichen Ressourcen dann zur Verfügung stehen, wenn sie benötigt werden. Die Constraint Programmierung wird hierbei verwendet, um die Planungskomponente des Reservierungsmanagers zu implementieren. Der Reservierungsmanager ermöglicht eine effizientere Auslastung der Cloud-Ressourcen.

1 Einleitung

Das Ressourcenmanagement ist für das Cloud Computing ein sehr komplexes Kernproblem, das auf verschiedenen Abstraktionsniveaus betrachtet werden kann. Für einen Cloud-Provider ist es sehr schwierig, eine vorausschauende dynamische und effiziente Ressourcenzuordnung zu organisieren. In diesem Kurzbericht wird ein Manager vorgestellt, der Reservierungen von Ressourcen verwaltet und eine Ressourcenplanung ermöglicht. Für einen Cloud-Provider besteht durch die Verwendung dieses Managers die Möglichkeit, die Fertigstellung eines Dienstes zu einem bestimmten Zeitpunkt zu garantieren. Um solche zeitliche Garantien geben zu können, muss der Cloud-Provider sicher sein, dass die erforderlichen Ressourcen der Cloud in dem relevanten Zeitraum auch verfügbar sind. Durch die Verwendung des Reservierungsmanagers kann eine solche Ressourcenverfügbarkeit bereits vor der Annahme des gewünschten Dienstes überprüft werden. Außerdem kann der Cloud-Provider mittels dieses Managers auf Kundenwunsch auch garantieren, dass die Kundendaten gewünschte örtliche Regionen nicht verlassen.

Die Verwendung eines Reservierungsmanager ist nur sinnvoll, wenn für Dienste Fertigstellungszeiten garantiert werden sollen und wenn diese Dienste für eine längere Zeit einen größeren Umfang an Cloud-Ressourcen benötigen. Eine vorausschauende Reservierung von Ressourcen wird in [FBF12] unter der Anwendung eines spieltheoretischen Ansatzes diskutiert. Techniken der stochastischen Integer-Programmierung werden in [LG2010] verwendet, um ein auf SLA (Service Level Agreement) basierendes Ressourcen-Scheduling zu optimieren. Eine Scheduling-Strategie, die Reservierungen für priorisierte Jobs durchführt und ein dynamisches Scheduling realisiert, wird in [Bar13] beschrieben. In [SS12] wird ein Grundalgorithmus für eine dynamische Ressourcenzuordnung vorgeschlagen, der Vertragsvereinbarungen (SLA) und Risikoanalyse berücksichtigt. Dieser Grundalgorithmus enthält auch eine Planung von Ressourcen-Reservierungen und Rescheduling auf

[1] Fraunhofer FOKUS, 10589 Berlin, {hans-joachim.goltz, armin.wolf}@fokus.fraunhofer.de

Basis der SLA. Ein temporales Knapsack-Problem und deren Lösungsmöglichkeiten werden in [BF05] diskutiert. Ein Planungsproblem von Ressourcen-Reservierungen kann auch als ein temporales Knapsack-Problem betrachtet werden.

Für die Implementierung des hier vorgestellten Reservierungsmanagers wird softwaretechnisch die constraintbasierte Programmierung mit globalen Constraints verwendet. Die constraintbasierte Programmierung ist eine in der Praxis bewährte Technologie für die Lösung diskreter komplexer Probleme, insbesondere für das Lösen von Scheduling- und Ressourcenzuordnungsproblemen. Speziell wurde die am Fraunhofer Institut entwickelte Constraint-Solver-Bibliothek *firstCS* ([Wo12]) in die Implementierung des Reservierungsmanagers integriert.

Der beschriebene Reservierungsmanager wurde innerhalb des Projektes "EASI-CLOUDS" entwickelt, das ein Projekt des ITEA2 Programms ist (siehe auch [FL14]). Der deutsche Teil des Projektes wurde durch das Bundesministerium für Bildung und Forschung (BMBF) unter der Nummer 01 IS 11021 gefördert.

2 Reservierungsmanager

Die Hauptaufgabe des Reservierungsmanagers besteht darin zu überprüfen, ob für jeden Dienst mit vertraglich vereinbarten zeitlichen und örtlichen Bedingungen der Dienstrealisierung die erforderlichen Ressourcen dann zur Verfügung stehen, wenn sie benötigt werden. Der Reservierungsmanager befürwortet die Annahme eines Dienstes nur dann, wenn zu erwarten ist, dass die erforderlichen Ressourcen unter den gewünschten zeitlichen und örtlichen Einschränkungen auch in dem notwendigen Umfang zur Dienstrealisierung genutzt werden können. Die Planungskomponente des Managers bestimmt unter Berücksichtigung der Ressourcenverfügbarkeit den Zeitpunkt, wann ein Dienst startet und kann solche Startzeitpunkte unter Einhaltung aller Bedingungen auch verschieben.

Die Verwendung eines Reservierungsmanager ist nur sinnvoll, wenn für Dienste Fertigstellungszeiten garantiert werden sollen und wenn diese Dienste für eine längere Zeit einen größeren Umfang an Cloud-Ressourcen benötigen. Da diese Bedingungen allgemein nicht alle zu realisierende Dienste erfüllen, kann ein solcher Reservierungsmanager nur dann sinnvoll eingesetzt werden, wenn ein festgelegter Anteil der vorhandenen Ressourcen des Providers für Reservierungen zur Verfügung stehen. Der andere Anteil der Ressourcen kann dann durch Dienste ohne Reservierung und als Reserve genutzt werden.

Der Reservierungsmanager ist eng mit dem SLA-Manager verbunden. Bevor der SLA-Manager einen Kunden-Auftrag annimmt, muss er den Reservierungsmanager fragen, ob die erforderlichen Ressourcen mit den gewünschten zeitlichen und örtlichen Restriktionen zur Verfügung stehen oder unter welchen zeitlichen Bedingungen die erforderlichen Ressourcen verfügbar sind. Der Reservierungsmanager wird nur aktiviert, wenn er eine Anfrage oder Nachricht erhält. Das definierte Interface basiert auf der Kommunikation mit vorgegebenen XML-Termen. Da der Reservierungsmanager eng mit dem SLA-Manager verbunden ist und diese beiden Manager in Java implementiert sind, wurde das Interface

des Reservierungsmanagers so realisiert, dass die Kommunikation sowohl mittels XML-Termen als auch direkt innerhalb von Java erfolgen kann.

Durch eine in XML spezifizierte Ressourcenbeschreibung wird festgelegt, welche Ressourcen mit welchen Eigenschaften der Reservierungsmanager verwalten kann. Da der Reservierungsmanager nur Aussagen zur Verfügbarkeit von Ressourcen und keine Zuordnung konkreter Ressourcen realisiert, erfolgt die Ressourcenbeschreibung in einer abstrahierten Form, bei der vor allem Kapazitäten von Ressourcenklassen relevant sind. Dabei werden Ressourcen mit gleichen Eigenschaften zu Ressourcenklassen zusammengefasst. Somit kann ein Cloud-Provider auch festlegen, für welchen Teil seiner Ressourcen eine Ressourcenplanung erfolgt.

Eine Anfrage, die Verfügbarkeit von Ressourcen zu überprüfen oder Ressourcen zu reservieren, enthält Spezifikationen von kombinierten Ressourcen mit der jeweils gewünschten Anzahl. Die Spezifikation einer kombinierten Ressource definiert die zusammenhängenden Ressourcen, die für die Realisierung eines Dienstes erforderlich sind (zum Beispiel, die Ressourcen, die für eine Virtuelle Maschine benötigt werden). Eine solche Spezifikation enthält die Spezifikationen der einzelnen Ressourcen (zum Beispiel: Netzwerk, Festplattenspeicher, CPU) und Zeit-Constraints für die zeitlichen Beziehungen dieser Ressourcen untereinander. Die kombinierten Ressourcen werden von einander als unabhängig betrachtet. Wenn die gewünschten Kapazitäten der erforderlichen Ressourcen einer Anfrage in dem gewünschten Zeitintervall nicht verfügbar sind, dann kann der Reservierungsmanager nur eine Reduzierung der gewünschten Anzahl der kombinierten Ressourcen vorschlagen. Eine kombinierte Ressource ist aus Sicht des Reservierungsmanagers immer eine zusammenhängende Einheit. Die Einführung der Definition von kombinierten Ressourcen war ein wichtiges Hilfsmittel, um ein effizientes Management von Ressourcen realisieren zu können.

Bei einer Anfrage zur Reservierung von Ressourcen werden durch den Reservierungsmanager Startzeiten für jede realisierbare kombinierte Ressource zurückgegeben. Reservierungszeiten von Ressourcen können fest oder nicht fest sein. Wenn Reservierungszeiten nicht fest sind, dann können diese durch den Reservierungsmanager noch auf später unter Einhaltung aller zeitlichen Bedingungen verschoben werden. Da der Reservierungsmanager nur auf Anfragen reagiert und Zeitverschiebungen nicht automatisch mitteilt, muss ein Dienst, der noch nicht feste Reservierungszeiten erhalten hat, vor dem Start den Reservierungsmanager nach diesen Zeiten fragen.

Um zu vermeiden, dass parallel die gleichen Ressourcen für verschiedene Dienste für die gleiche Zeit reserviert werden, wurde das Reservierungsmanagement sequentiell implementiert. Zur gleichen Zeit wird immer nur die Anfrage eines Dienstes bearbeitet. Deshalb kommen alle Anfragen in eine Warteschlange und werden sequentiell bearbeitet.

3 Implementierung mittels constraintbasierter Programmierung

Die constraintbasierte Programmierung wurde für die Implementierung des Reservierungsmanagers, insbesondere für die enthaltene Planungskomponente, verwendet. Die cons-

traintbasierte Programmierung ist besonders dann geeignet, wenn komplexe Zeit- und Ressourcenplanungsprobleme deklarativ durch Entscheidungsvariablen und Randbedingungen (Constraints) modelliert werden können. Den Variablen sind jeweils Mengen von möglichen Werten zugeordnet und kennzeichnen typischerweise auch mögliche alternative Entscheidungen. Constraints repräsentieren relationale Beziehungen zwischen Variablen. Durch Constraints wird ausgedrückt, welche Bedingungen zulässige Belegungen der Variablen erfüllen müssen, und beschränkt somit die Wertekombinationen der Variablen. Eine Lösung eines Constraintproblems ist eine Zuordnung aller Variablen zu Werten aus ihren Wertebereichen, so dass alle Constraints erfüllt sind.

Für die Implementierung des vorgestellten Reservierungsmanagers wurde die am Fraunhofer Institut entwickelte Constraint-Solver-Bibliothek *firstCS* verwendet. Diese Bibliothek stellt dem Nutzer die notwendigen Konzepte zur Verfügung, um constraintbasierte Optimierungsprobleme über endlichen Wertebereichen zu modellieren und zu lösen. Der Anwendungsfokus von *firstCS* ist constraintbasiertes Scheduling und Ressourcenzuordnung. Im Folgenden wird kurz charakterisiert, wie der Reservierungsmanager mittels dieser Bibliothek realisiert wurde.

Die zeitlichen Constraints können üblicherweise durch Gleichungen und Ungleichungen modelliert werden. Globale *cumulative* Constraints werden verwendet, um Kapazitätsüberschreitungen der Ressourcen zu vermeiden. Durch ein kumulatives Constraint kann gesichert werden, dass zu keinem Zeitpunkt die gegebene Kapazität einer Ressource überschritten wird. Das *cumulative* Constraint wurde ursprünglich in [AB93] eingeführt, um Scheduling- und Platzierungsprobleme zu lösen.

Bei einer Anfrage zur Reservierung von Ressourcen für einen Dienst existieren im Allgemeinen verschiedene Möglichkeiten der Ressourcenzuordnung einer angefragten Ressourcenart. In einem solchen Fall kann die Vermeidung von Kapazitätsüberschreitung durch die Verwendung von alternativen kumulativen Constraints modelliert werden. Dies ist eine Verallgemeinerung des kumulativen Constraints bezüglich der Möglichkeit einer alternativen Ressourcenzuordnung. Ein solches Constraint ist in der verwendeten Constraint-Solver-Bibliothek *firstCS* implementiert und wurde auch für die Realisierung des Reservierungsmanagers verwendet. Die Implementierung des alternativen kumulativen Constraints in *firstCS* basiert auf einer Verallgemeinerung der in [WS05] dargestellten Ergebnisse.

Das Erzeugen von alternativen kumulativen oder kumulativen Constraints ist nur dann für eine Ressource erforderlich, wenn die Möglichkeit besteht, dass die Kapazitätsgrenzen überschritten werden. Für jede angefragte Quantität einer Ressource überprüft deswegen der Reservierungsmanager, ob für diese Ressource in dem relevanten Zeitintervall die angefragte Quantität ohne Einschränkung zur Verfügung steht. Nur wenn dies nicht der Fall ist, werden die genannten Constraints für diese Ressource erzeugt.

Für die Lösungssuche wurde eine anwendungsspezifische Strategie entwickelt, die wesentlich effizienter als die allgemeine Standardstrategie zur Lösung solcher constraintbasierter Optimierungsprobleme ist und spezifische Eigenschaften der Anwendung nutzt. Eine Anfrage zur Verfügbarkeit oder Reservierung von Ressourcen enthält die Spezifikation einer

gewünschten Anzahl von kombinierten Ressourcen. Eine kombinierte Ressource ist nur dann verfügbar (und somit reservierbar), wenn alle Ressourcen, die zu dieser gehören, auch verfügbar sind und alle zugeordneten Constraints erfüllt sind. Deswegen versucht die Planungskomponente für die spezifizierte kombinierte Ressource schrittweise konsistente Zuordnungen zu finden, bis die gewünschte Anzahl erreicht ist. Da jede dieser kombinierten Ressourcen die gleichen Ressourcen und zeitlichen Constraints enthält, wird bei der Lösungssuche Backtracking nur für die Ressourcenzuordnung innerhalb einer kombinierten Ressource erlaubt. Wenn es keine Lösung für eine kombinierte Ressource mehr gibt, dann kann die Lösungssuche abgebrochen werden und die bereits vorher zugeordnete Anzahl dieser kombinierten Ressource ist das Ergebnis. Wenn die gewünschte Anzahl von kombinierten Ressourcen nicht erreicht werden kann, dann wird versucht eine Lösung zu finden, bei der bereits reservierte kombinierte Ressourcen geplant werden, falls dies erlaubt und möglich ist. Wenn eine Anfrage verschiedene kombinierte Ressourcen enthält, dann wird dies als eine Folge von Anfragen mit gleichen kombinierten Ressourcen betrachtet, wobei die Reihenfolge durch die vom Nutzer vorgegebene Priorität bestimmt wird.

4 Schlussfolgerungen

Der vorgestellte Reservierungsmanager verwaltet Reservierungen von Ressourcen und kann durch einen Cloud-Provider verwendet werden, um die Fertigstellung von Diensten zu bestimmten Zeitpunkten garantieren zu können. Dies ist aber nur für solche Dienste sinnvoll, die für eine längere Zeit einen größeren Umfang an Cloud-Ressourcen benötigen. Um solche zeitliche Garantien geben zu können, muss der Cloud-Provider sicher sein, dass die erforderlichen Ressourcen der Cloud in dem relevanten Zeitraum auch verfügbar sind. Durch die Verwendung des Reservierungsmanagers kann eine solche Ressourcenverfügbarkeit bereits vor der Annahme des gewünschten Dienstes überprüft werden. Für die Nutzung eines solchen Reservierungsmanagers ist es notwendig, dass der Cloud-Provider einen Teil der Cloud-Ressourcen für das Reservierungsmanagement zur Verfügung stellt. Durch eine erste prototypische Implementierung des Reservierungsmanagers und deren Integration in ein Cloud-Projekt konnte nachgewiesen werden, dass das vorgestellte Konzept erfolgreich realisiert werden kann. Für diese Implementierung wurde softwaretechnisch die constraintbasierte Programmierung mit globalen Constraints verwendet.

Literaturverzeichnis

[AB93] Aggoun, A., Beldiceanu, N.: Extending CHIP in order to solve complex scheduling and placement problems. J. Mathematical and Computer Modelling 17(7), 57–73, (1993).

[BF05] Bartlett, M., Frisch, A.M., Hamadi, Y., Miguel, I., Tarim, S.A., Unsworth, C.: The Temporal Knapsack Problem and Its Solution. In: R. Bartak and M. Milano (Eds.), CPAIOR 2005, LNCS, vol. 3524, pp. 34–48, Springer, Berlin, Heidelberg (2005).

[Bar13] Barnabas, D.: Resource Reservation in the Cloud based on Infrastructure as a Service (IaaS). Research Journal of Computer Systems Engineering RJCSE, Vol. 04; Special Issue; (2013).

[FL14] Fiehe, C., Litvina, A., Tonn, J., Wu, J., Scheel, M., et al.: Building a Medical Research Cloud in the EASI-CLOUDS Project. In: Proceedings 6th International Workshop on Science Gateways, IWSG 2014, IEEE Xplore Digital library (2014).

[FBF12] Funke, D.,Brosig, F., M.Faber, M.: Towards Truthful Resource Reservation in Cloud Computing. In: Proc. of 6th International ICST Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS), pp. 253–262 (2012).

[LG2010] Li, Q., Guo, Y.: Optimization of Resource Scheduling in Cloud Computing. In: Proc. of 12th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, IEEE (2010).

[SW13] Sabharwal, N., Wali, P.: Cloud Capacity Management. Apress (part of Springer Science+Business Media), New York (2013).

[SS12] Siddesh, G.M., Srinivasa, K.G.: SLA - Driven Dynamic Resource Allocation on Clouds. In: Thilagam, P.S. et al. (Eds.), ADCONS 2011, LNCS, vol. 7135, pp. 9–18, Springer, Berlin, Heidelberg (2012).

[Wo12] Wolf, A.: firstCS - New Aspects on Combining Constraint Programming with Object-Orientation in Java. KI – Künstliche Intelligenz 26(1), 55–60 (2012).

[WS05] Wolf, A., Schlenker, H.: Realising the Alternative Resources Constraint. In: D. Seipel et al. (Eds.): INAP/WLP 2004, LNAI, vol. 3392, pp. 185–199. Springer Berlin Heidelberg (2005).