Auto-tuning in theorie en praktijk

Peter Sap – peter@petersap.nl

Inleiding

Auto-tuning is een faciliteit die ervoor moet zorgen dat een RDBMS nooit performanceproblemen kent of deze terugbrengt tot een acceptabele vorm. Het zal duidelijk zijn dat auto-tuning alleen al daarom een enorm breed onderwerp is: van het automatisch herschrijven van inefficiënte SQL naar een betere vorm, het aanpassen van configuratie parameters op het moment dat het nodig is, het zonodig reorganiseren van tabellen, etc, etc. In dit artikel wordt slechts een deel er uitgelicht, namelijk het automatisch instellen van configuratie parameters, met een uitstapje naar indexen en het plaatsen van tabellen op harde schijf. In een tweeluik worden achtereenvolgens enkele onderliggende theorieën en voorbeelden uit de praktijk beschreven.

De noodzaak van auto-tuning

Voor een mission-critial informatiesysteem is een goede performance, die te allen tijde kan worden gegarandeerd, van levensbelang voor een onderneming. Grote organisaties hebben daarom vaak een of meerdere ervaren DBA's in dienst, die geacht worden de performance te meten en te bewaken. Goede en ervaren DBA's zijn schaars en vormen een behoorlijke kostenpost voor de IT afdeling. Total cost of ownership wordt voor een groot deel bepaald door de kosten van personeel en alleen al daarom is een zelf tunend RDBMS uitermate welkom. Daarnaast kan handmatige tuning vaak niet goed inspelen op plotselinge wijzigingen in de workload. Immers, pas nadat is geconstateerd dat de performance is teruggelopen worden wijzigingen doorgevoerd. Het is inherent aan deze manier van werken dat de DBA altijd achter de feiten aanloopt.
In de huidige tijd van E-commerce wordt auto-tuning nog belangrijker dan voorheen:

- De workload voor een RDBMS kan enorm fluctueren doordat een website een grote fluctuatie in aantallen bezoekers heeft. Daarnaast is een 24 uurs beschikbaarheid, gekoppeld aan goede prestaties, essentieel.
- Een webdienst met een slechte performance zal niet snel opnieuw door een teleurgestelde bezoeker worden bezocht.
- Sommige analisten schatten de geleden schade door het niet beschikbaar zijn van een grote webdienst in op bedragen in de orde van duizenden dollars per minuut, puur door verlies aan klanten en negatieve invloed op de positie in de markt. Een haperende webdienst kan dezelfde gevolgen hebben.

Deel 1: de theorie van auto-tuning

Auto-tuning is oud

Kennis van de theorie over een bepaald onderwerp is altijd nuttig, en bij auto-tuning is dat niet anders. Gewapend met die kennis kan dan beter worden ingeschat of de leverancier van een RDBMS echte auto-tuning biedt, of slechts een stuk gereedschap dat het leven van de DBA gemakkelijker moet maken.
Gerhard Weikum et. al. publiceerde in 1994 het artikel "The Comfort Automatic Tuning Project". Zijn team had gedurende vier jaar met een prototype onderzocht of auto-tuning haalbaar was en naar een mogelijke oplossingsrichting gezocht. De conclusie was dat auto-tuning gerealiseerd kan worden maar er zou nog veel onderzoek moeten plaatsvinden. De onderliggende theorie van het prototype was gebaseerd op een terugkoppelingsmechanisme.
Een andere vorm van auto-tuning werkt met applicaties of transacties die, afhankelijk van de gewenste performance, in een bepaalde klasse kunnen worden ondergebracht. Aan een klasse kan dan een bepaalde prioriteit worden gekoppeld. Deze vorm heet 'Goal-oriented resource allocation'.
Een derde variant is de zogenaamde broker methode waarbij, zoals op een aandelenbeurs, een proces binnen het RDBMS een bod moet uitbrengen om de beschikking te kunnen krijgen over bijvoorbeeld een deel van het interne geheugen. De hoogste bieder en daarmee de meest belanghebbende, krijgt dan de beschikking over de benodigde systeemcomponent. In dit artikel wordt deze vorm niet verder beschreven.

De werking van auto-tuning volgens Weikum

De werking van het hierboven genoemde Comfort prototype was gebaseerd op een terugkoppelingsmechanisme bestaande uit drie delen:

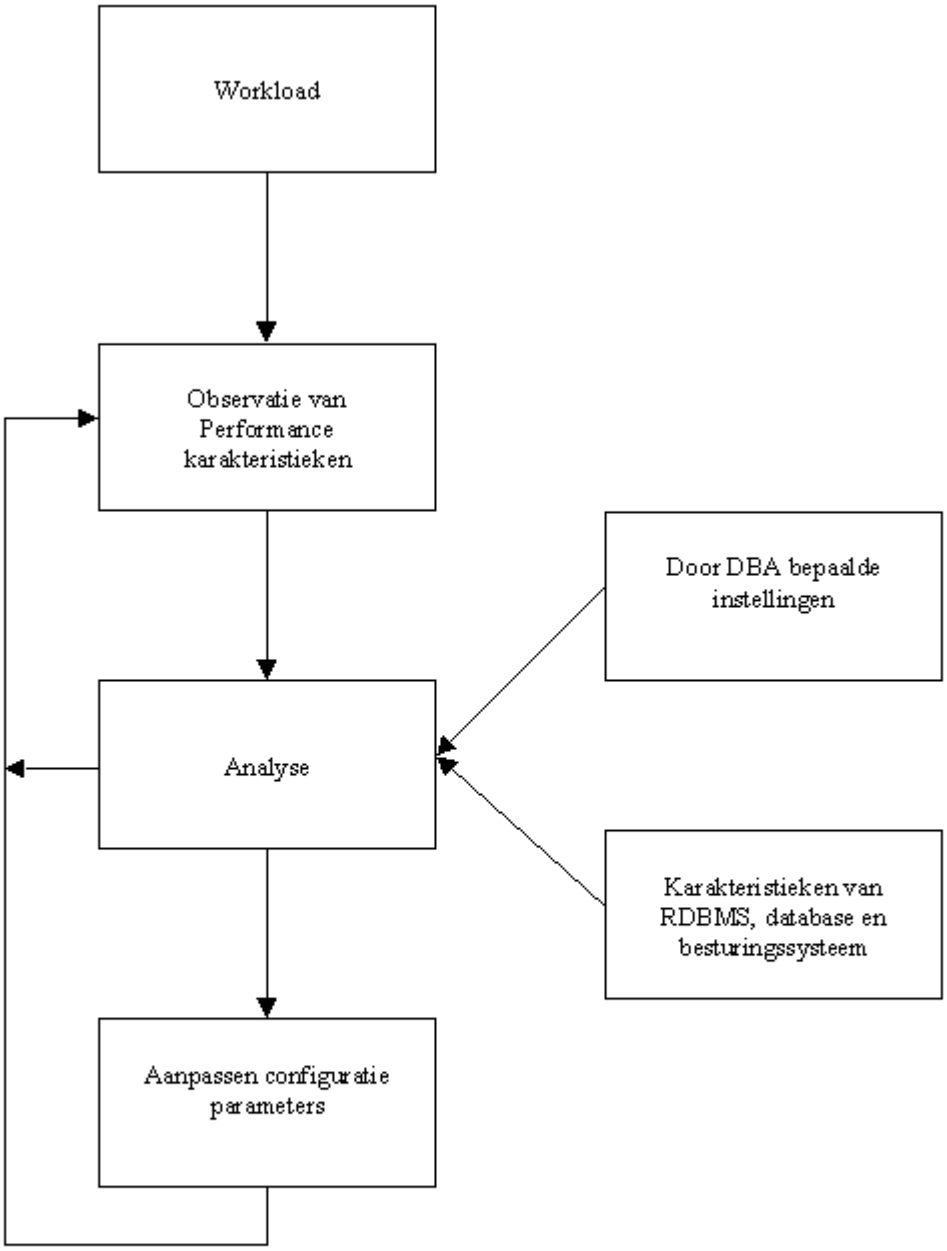
1. De workload voor een RDBMS wordt bijgehouden en de performance karakteristieken gemeten (Observeer)
2. Er wordt geëvalueerd of een wijziging in de configuratie de performance ten goede zou kunnen komen (Voorspel).
3. De configuratie wordt zonodig aangepast (Reageer).

Na het aanpassen van de configuratie wordt de workload opnieuw bijgehouden in het observatiegedeelte.

In dit ogenschijnlijk effectieve mechanisme zitten echter nog een aantal lastige problemen. In het observatiedeel moet worden vastgesteld wat precies de te meten karakteristieken van een workload zijn. Responsetijd of throughput alleen kan niet voldoende zijn aangezien er geen inzicht in het gedrag van de onderliggende RDBMS componenten wordt verkregen. Het gedrag van bijvoorbeeld het buffergeheugen moet bekend zijn om te kunnen voorspellen of een aanpassing hierin de performance kan verbeteren. Er zal dus een behoorlijk aantal karakteristieken moeten worden gemeten en bijgehouden om een complete observatie te kunnen uitvoeren. Met de huidige stand van de techniek is dit mogelijk; de overhead van de meting zal acceptabel zijn.

Als vanuit de observatie wordt besloten dat er een configuratieaanpassing moet plaatsvinden moet worden voorkomen dat het systeem instabiel wordt of dat een aanpassing een ongewenste bijwerking heeft. Om dit te kunnen realiseren zijn wiskundige modellen nodig die de interactie tussen alle verschillende parameters beschrijven en het doorrekenen van die modellen moet voldoende snel kunnen gebeuren. Helaas moet worden geconstateerd dat zo'n model er op dit moment nog niet is. De modellen die er zijn werken slechts voor een beperkt aantal parameters.

Tenslotte, in het deel dat de configuratie dynamisch aanpast moet het mogelijk zijn om alle gewenste instellingen te kunnen wijzigen zonder dat een herstart van de server nodig is, of dat de workload er negatief door wordt beïnvloed. In een aantal situaties is dit nog niet mogelijk.

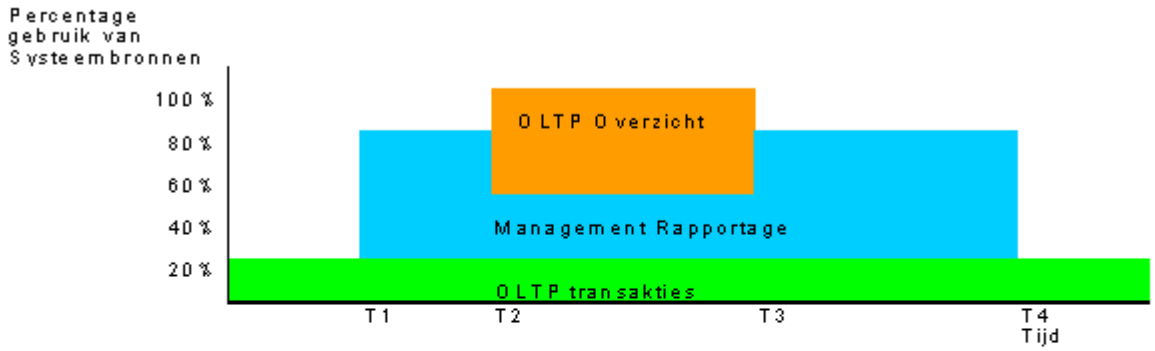


Figuur 1: Schematische weergave van een terugkoppelingsmechanisme waarbij de workload wordt geobserveerd en vervolgens wordt geanalyseerd. Bij de analyse wordt rekening gehouden met de kenmerken van het RDBMS, de database en het gebruikte besturingssysteem. Tevens kan de DBA invloed uitoefenen op het beslissingsproces door op te geven dat sommige parameters niet mogen worden aangepast. Afhankelijk van de analyse wordt er al dan niet een aanpassing in de configuratie gemaakt.

Goal-oriented resource allocation

Vanaf een hoger niveau beschouwd, kent het hierboven beschreven terugkoppelingsmechanisme een ander groot nadeel. Een database toepassing die zowel OLTP en management rapportages op hetzelfde moment moet kunnen ondersteunen, zou getuned worden zonder rekening te houden met de belangen van de verschillende toepassingen. De OLTP verwerking verlangt een snelle responsetijd terwijl dat voor de rapportages geheel anders kan liggen. Bij goal-oriented resource allocation worden transacties ingedeeld in klassen waaraan bepaalde doelen worden gesteld. Een voorbeeld daarvan is

een responsetijd van een seconde voor de OLTP transacties, tien seconden voor een overzicht vanuit de OLTP toepassing en 'zo snel mogelijk' voor management rapportage.
Tuning zal nu steeds plaatsvinden door rekening te houden met de belangen van een klasse. Zodra de OLTP transacties niet binnen een seconde kunnen worden uitgevoerd wordt automatisch de configuratie bijgesteld ten nadele van de klassen die minder belangrijk zijn. Aan de andere kant, als de OLTP transacties wel binnen een seconde worden verwerkt wordt niet verder getuned om deze nog sneller te laten verwerken, maar worden de resterende resources vrijgemaakt ten gunste van de overige klassen. Om onnodig veel aanpassingen tegen te gaan zal met een tolerantie worden gewerkt.
Ook hier wordt met een soortgelijk terugkoppelingsmechanisme gewerkt zoals dat door Weikum is beschreven (Observeer, Voorspel en Reageer), maar dat gebeurt nu steeds binnen een klasse.



Tijdspad van het verloop van enkele toepassingen en hoe goal-oriented resource allocation hier mee omgaat. De OLTP transacties hebben de hoogste prioriteit en vragen continue om 20% van de beschikbare systeembronnen (Groene balk) Op tijstip T1 wordt een langdurige management rapportage aangevraagd maar dit kan samen met de OLTP toepassing ruim binnen de beschikbare capaciteit worden opgelost (Blauwe balk). De systeembronnen worden niet boven de 80% belast. Op het moment dat een overzicht ten behoeve van de OLTP verwerking wordt opgestart op tijdstip T2 kan (Oranje balk), waarbij deze een hogere prioriteit heeft dan de managementrapportage, kan dit niet meer binnen de beschikbare systeembronnen worden uitgevoerd. De toegewezen systeembronnen voor management rapportage wordt dan teruggebracht om voorrang te verlenen. Als het overzicht klaar is op tijdstip T3 worden de bronnen weer vrijgegeven.

Asilomar

Eind 1998 kwam het zogenaamde Asilomar rapport uit waarin de noodklok over database research werd geluid. In dit rapport werden enkele wetenschappelijke uitdagingen voor het komende decennium opgesomd en auto-tuning was er een van. Immers, begin jaren negentig was al bekend dat total cost of ownership voor een groot deel bestaat uit personeelskosten en auto-tuning zou die kosten moeten terugdringen. Omdat vanuit de wetenschap nog geen bruikbaar model voor auto-tuning was aangedragen werd dit onderwerp hoger op de agenda geplaatst. Zonder dwingend te zijn werd voorgesteld om auto-tuning met wizards te laten werken. Dergelijke wizards moeten de workload op een server meten en zelf, waar nodig, de configuratie aanpassen.

Deel 2: auto-tuning in de praktijk

In dit deel worden voor DB2, Oracle, Sybase en Microsoft enkele gereedschappen beschreven die een vorm van auto-tuning bieden. Er wordt niet gepoogd een compleet en uitputtend overzicht per leverancier te bieden, maar een illustratie te geven hoe de verschillende RDBMS implementaties met auto-tuning omgaan. Overeenkomstige faciliteiten kunnen door verschillende leveranciers worden geboden, zonder dat het hier uitdrukkelijk wordt vermeld.

IBM DB2 Universal database

DB2 sluit mooi aan op de aanbevelingen van het Asilomar rapport en heeft een aantal wizards waarmee performance en tuning activiteiten kunnen worden uitgevoerd. Hiervan bieden de Governor Utility en het autoconfigure commando een vorm van auto-tuning.
Met het autoconfigure commando, of met de grafische interface die vanuit het Control Center kan worden opgeroepen, wordt na het opgeven van een aantal variabelen een aantal configuratie

parameters voorgesteld die direct kunnen worden toegepast op de database of in een script kunnen worden bewaard. De variabelen die moeten worden opgegeven hebben betrekking op onder meer het interne geheugen, soort workload voor de database en het aantal connecties. De wizard bepaald verder zelf de karakteristieken van de hardware zoals disks, besturingssysteem en omvang van database, tabellen en indexen. Volgens IBM zouden de berekende configuratieparameters een performance bieden die slechts enkele procenten trager is dan een handmatige tuning, uitgevoerd door experts van IBM. Interessant hierbij is dat de door autoconfigure berekende parameters een performance bieden die aanmerkelijk sneller is dan de standaard DB2 configuratie instellingen. Bij een bepaalde test, die in de literatuur wordt beschreven, was het echter nodig om één parameter handmatig aan te passen. De DBA is dus nog niet geheel buitenspel gezet.
De Governor utility werkt voor een groot deel volgens het hierboven beschreven 'goal-oriented resource allocation' algoritme. Met een aantal regels die door de DBA moeten worden opgegeven, kunnen acties worden gespecificeerd die moeten worden uitgevoerd als een applicatie een bepaalde limiet overschrijdt. Er kan onder meer een limiet worden gezet op het aantal verbruikte CPU seconden, het aantal locks die worden vastgehouden, het aantal geselecteerde rijen en de totale tijd dat een transactie actief is. De actie die moet worden uitgevoerd kan variëren van het wijzigen van de prioriteit, het beëindigen van een applicatie of verbinding met een gebruiker of een bepaalde klasse van applicaties voorrang verlenen.
Zoals gezegd, het mechanisme werkt met een uitgeklede vorm van 'goal-oriented resource allocation' en de beslissingen worden feitelijk door de DBA gemaakt die de regels opstelt.

Oracle 9i

Oracle Expert, een onderdeel van Enterprise Manager, is de belangrijkste wizard waarmee auto-tuning wordt aangeboden. Hiermee kunnen onder meer de instance parameters worden getuned, intern hergebruik van SQL worden gecontroleerd en een uitgebreide toegangspad analyse worden uitgevoerd. Het gehele proces voor de wizard is gebaseerd op het verzamelen van een historie van acties die op de database zijn uitgevoerd, aangevuld met kenmerken over de workload overeenkomstig auto-configure van DB2. Al deze gegevens kunnen vervolgens worden geanalyseerd door Oracle Expert. Vanuit de analyse worden de aanbevelingen zichtbaar gemaakt en kan door de DBA worden besloten deze te accepteren of een of meerdere instellingen niet te wijzigingen. Als niet alle aanbevelingen in hun geheel worden overgenomen moet de analyse opnieuw worden uitgevoerd om te controleren of de geaccepteerde parameters consistent zijn.
Omdat veel tuning parameters een onderlinge samenhang hebben zijn er regels binnen Oracle Expert die de samenhang beschrijven. De DBA kan deze regels aanpassen zodat de aanbevelingen vanuit de wizard beter aansluiten bij een specifieke situatie.
Bij het aanpassen van de instance parameters moeten eerst een aantal metingen zijn uitgevoerd en doorgerekend door Oracle Expert alvorens een resource kan worden teruggeschroefd. Hiermee wordt voorkomen dat één meting er toe zou leiden dan een bepaalde resource te ver wordt teruggeschroefd en de performance sterk terug zou lopen. Vanzelfsprekend moeten de metingen worden uitgevoerd op het moment dat de database zwaar wordt belast.

Microsoft SQL-Server

In Microsoft SQL-server is een heel anders soort wizard aanwezig, namelijk de index tuning wizard. Met behulp van een bepaalde workload die kan worden verzameld met SQL-profiler, kan de index tuning wizard analyseren welke indexen benodigd zijn om de workload zo optimaal mogelijk te verwerken. Om te voorkomen dat de aanbevolen indexen een te groot aantal kolommen gaat beslaan is er een limiet, die door de DBA kan worden ingesteld. Tevens kan er worden opgegeven wat het te verwachte aantal rijen voor een tabel in de toekomst zal zijn. Dit is een handige optie die kan worden gebruikt voordat een systeem in productie wordt genomen.
Ook kan een rapport worden opgevraagd waarmee de overbodige, of weinig gebruikte indexen, worden gesignaleerd.
Volgens opgave van Microsoft kan met de meest uitgebreide analyse een performancewinst van bijna 80% worden behaald. Er wordt echter niet vermeld wat de uitgangssituatie voor de test was.

Sybase

Met Adaptive Server Enterprise zit Sybase in de achterhoede op het gebied van auto-tuning. Vanuit sp_sysmon, het gereedschap waarmee performanceproblemen kunnen worden geanalyseerd, worden aanbevelingen gegeven om de instellingen van de server te verbeteren. Verder is er de Resource

Governor, qua functionaliteit te vergelijken met de Governor utility van DB2. Helaas is er niet de mogelijkheid om bij overschrijding van een bepaalde limiet de prioriteit van een applicatie tijdelijk te verlagen.
SQL Anywhere Studio, het kleine zusje van de Enterprise versie, staat echter bekend om zijn onderhouds- en DBA vrije werking. Met dit product wordt dan ook over 'zero-administration' gesproken.

De laatste ontwikkeling

Bij het aanmaken van een database wordt deze vaak gealloceerd over een aantal harde schijven en de tabel en index structuren worden daarbinnen niet gebonden aan een bepaalde harde schijf. De achterliggende gedachte is dat hierdoor I/O parallellisme wordt bereikt, uiteindelijk zullen alle schijven een evenredig deel van alle objecten bevatten en hierdoor wordt de belasting gedeeld. Op het moment dat een tabel een bepaalde hot-spot kent zal er een afwijking in parallellisme gaan ontstaan waardoor bepaalde harde schijven meer of minder worden belast. Zelfs voor een database met een gering aantal tabellen is het vaak zeer lastig voor een DBA om zelf een goede verdeling te maken over de verschillende schijven. Meestal zijn er teveel tabellen en te weinig harde schijven. Alleen voor een batchverwerking wordt vaak een verdeling gemaakt.
Recentelijk is er een techniek, genaamd Two Step Greedy, beschreven, waarmee een workload wordt geanalyseerd en aanbevelingen kunnen worden gedaan om bepaalde tabellen te binden aan een of meerdere harde schijven. Wellicht dat er in de nabije toekomst een nieuwe wizard voor de DBA bijkomt waarmee de verdeling van tabellen over de beschikbare harde schijven gemakkelijker kan worden uitgevoerd.

Een wizard is geen tovenaar

Uiteindelijk is het aantal wizards, of de manier waarop de informatie vanuit de wizards wordt getoond, niet echt maatgevend om te bepalen of een RDBMS een bruikbare vorm auto-tuning biedt. Immers, het doel van auto-tuning is 'zero-administration' of dat zo goed mogelijk benaderen. Wizards spelen daar natuurlijk een belangrijke rol in, maar het is ook belangrijk om te weten hoe een RDBMS omgaat met een onverwachte fluctuatie in de workload.
Toen in 1998 het Asilomar rapport uitkwam werd auto-tuning op de wetenschappelijke agenda voor het komende decennium geplaatst, en er werden wizards voorgesteld. We zijn inmiddels vijf jaar verder en er zijn wizards maar die vragen nog steeds om een behoorlijke interactie met de DBA. Er worden al een beperkt aantal configuratieparameters automatisch aangepast, zoals dat in Oracle 9i gebeurt, en in de toekomst kunnen we waarschijnlijk een verdere uitbreiding van auto-tuning verwachten.

Bronnen:

- G. Weikum et al, The Comfort Automatic Tuning project, Information Systems Vol. 19, No. 5, 1994
- K.P. Brown, Goal-oriented memory allocation in database management systems, 1995
- P. Bernstein et al, The Asilomar Report on Database Research, 1998
- S. Chaudhuri et al, Rethinking Database System Architecture, Proceedings of VLDB, 2000
- G. Weikum et al, Self-tuning database technology and information services, Proceedings of VLDB, 2002
- S. Agrawal et al, Automating Layout of Relational Databases, Proceedings of Data Engineering 2003

Peter Sap (peter@petersap.nl) is een senior database ontwikkelaar/DBA

Juni 2003