Universidade do Minho
Escola de Engenharia
Departamento de Informática

Daniel Carvalho da Rocha

Selecção de Hiper-Cubos com Base em Padrões de Exploração OLAP

Outubro de 2011

Universidade do Minho
Escola de Engenharia

Daniel Carvalho da Rocha

Selecção de Hiper-Cubos com Base em Padrões de Exploração OLAP

Tese de Mestrado
Mestrado em Engenharia Informática

Trabalho efectuado sob a orientação do
Professor Doutor Orlando Manuel de Oliveira Belo

Outubro 2011

Selecção de Hiper-Cubos com Base em Padrões de Exploração OLAP

Daniel Carvalho da Rocha

Dissertação apresentada à Universidade do Minho para obtenção do grau de Mestre em Engenharia Informática, área de especialização em Sistemas de Suporte à Decisão, elaborada sob orientação do Professor Doutor Orlando Manuel de Oliveira Belo.

2011

Aos meus familiares

Agradecimentos

Eu não poderia começar por preencher este espaço, em que me é permitido prestar um pequeno tributo às pessoas que eu considero fundamentais e que definiram o caminho traçado para a obtenção de grau de Mestre em Engenharia Informática, sem dar um especial agradecimento à minha família. Obrigado pelo cuidado, obrigado pelas instruções de enorme valor que sempre me fizeram incutir, obrigado por fazerem de mim quem sou e obrigado por serem como sois. Eu tento que o meu agradecimento seja feito diariamente e de forma mais útil, do que endereçar um simples texto com frases emocionantes. Por isso, à minha família quero dizer muito obrigado.

De seguida, tenho que agradecer ao sistema de ensino, que fiz parte ao longo destes anos e que muito tem feito por se tornar cada vez melhor. Se hoje tenho condições de obter o grau de Mestre é porque ao longo da minha vida de aluno encontrei pessoas que de facto souberam cativar-me para os estudos, que tentaram subir o meu nível conhecimentos e capacidades, que transformou uma criança, num ser com valores e com vontade de continuar a apreender.

Obrigado à Universidade do Minho, por terem tornado estes cinco anos muito especiais e únicos, obrigado pelas recordações que carregarei sempre comigo e pelo contributo que representam para o meu desenvolvimento como ser humano. Esta instituição representa muito para mim. Fico feliz e com um sentimento de sorte, quando recordo o momento em que escolhi esta universidade para efectuar a minha formação como Engenheiro Informático. Das muitas coisas que aprendi na minha passagem nesta academia, foi que as universidades, as escolas, os departamentos, os cursos, as unidades curriculares, são palavras que simbolizam conjuntos de pessoas e as pessoas que conheci e que constituem esta universidade revelaram-se muito importantes para me tornar na pessoa que sou.

Quero agradecer ao Departamento de Informática e a todos que tornam os cursos de LEI e MEI realmente exigentes, que dá orgulho frequentar, cursos desafiantes, de enorme qualidade e prestígio. Assim gostaria de dar um especial agradecimento ao meu orientador Professor Doutor Orlando Manuel de Oliveira Belo, que nos últimos dois anos me cativou para estudar a fundo a área relativa aos Sistemas de Suporte à Decisão, que se revelou para mim como sendo de enorme interesse. A todos, obrigado pelas instruções, obrigado por me terem tornado Mestre em Engenharia Informática.

Resumo

Selecção de Hiper-Cubos com Base em Padrões de Exploração OLAP

Na literatura do domínio do processamento analítico de dados facilmente se podem encontrar métodos e soluções que respondem ao problema de selecção de vistas multidimensionais no processo de implementação de um cubo OLAP. Uma forma que se evidencia como sendo extremamente vantajosa, é a de fazer a selecção baseada em critérios que se apoiem essencialmente nos conteúdos que são consultados sobre o cubo de dados ao longo das sessões de consulta OLAP. As principais vantagens que advêm desta monitorização, está relacionada com a possibilidade de efectuar correspondências rigorosas com a informação em que os agentes de decisão mais se apoiam para efectuar as suas tomadas de decisão. Ao ser feita a identificação da informação que se evidencia como sendo a mais relevante, ou pelo menos a mais frequentemente consultada, várias ilações se podem retirar, como, por exemplo, a definição de perfis de utilização, a expressão de preferências, a identificação de metodologias de trabalho, ou então a definição de processos que procurem construir cubos *iceberg* com forte probabilidade de explorações futuras sobre o cubo. Este último aspecto constitui, basicamente, o trabalho desta dissertação. Ao se efectuar a materialização dos conteúdos mais pesquisados no servidor OLAP, obtém-se uma melhor performance ao nível do servidor, uma vez que o preparamos antecipadamente com os dados que mais vezes são solicitados, reduzindo assim o número de vezes que seria necessário recorrer ao *data warehouse* para retornar os resultados pretendidos por uma dada as *query* multidimensional. Em termos gerais, neste trabalho de dissertação, desenvolveu-se um estudo detalhado acerca das ideias e práticas que levam ao desenvolvimento de um dado método de selecção, que seja capaz de indicar de forma precisa as partes de um cubo que são mais utilizadas, sugerindo com base nessa informação uma nova estrutura para o cubo em questão que utilize menos recursos computacionais, nomeadamente espaço em disco e tempo de processamento.

Abstract

Selection of Hiper-Cubes Based on OLAP Exploration Standards

In the literature, we can easily find methods and solutions that solve the problem of pick a set of multidimensional views in the implementation of a data cube process. One way that is shown to be extremely advantageous, is to make this selection based on criteria directly related with the contents that are searched on the data cube, along the OLAP query sessions. The main advantage that becomes with this monitoring process is the ability to make accurate matches with the information that decision-makers really are interested. With the identification of the information that is characterized as being the most researched, several conclusions and utilities can be made, such as setting profile users, find expressions of preferences, identify methods of work, or defining processes that build iceberg cubes with a strong probability of further explorations, subject that is discussed in this master thesis. Better performance can be developed in the server if we materialize only the most researched content on the OLAP server. The server is prepared with the data that is more times requested and then the number of times that is needed to exploit the data warehouse is reduced. In this master thesis is produced a study who combines ideas and practices that lead to the development of a selection method that makes a very precise indication of the contents to be selected, making a constant control of the multidimensional queries made on the data cube. Then we can identify which parts of the cube have the priority to be materialized. With this resolution, we can provide a more effective utilization for the community of users.

Índice

Índice de Figuras

Índice de Tabelas

Índice de Algoritmos

Capítulo 1

Introdução

1.1 Enquadramento de Trabalho

Como se pôde verificar ao longo dos últimos anos, o actual estado dos mercados, independentemente da sua natureza, encontra-se de tal forma competitivo que é difícil desenvolver posições fortes no seu seio por períodos consideráveis de tempo. As empresas e os grupos económicos têm uma grande necessidade de informatizar os seus processos para que, assim, obtenham uma melhor organização, eficiência de comunicação e recolha da informação, que lhes permita definir concretas formas de acção mais efectivas no dia-a-dia dos seus negócios [Connolly & Begg 2001]. De facto cada vez mais são gerados dados sobre todos os tipos de recursos utilizados pelas empresas ao longo das suas actividades. Isto faz com que, em muitos cenários de aplicação, possam ocorrer inúmeros casos indesejáveis de desorganização da informação quando se analisa uma organização como um todo.

Assim, quando os agentes de decisão empresariais pretendem obter valores acerca dos seus negócios, cruzando dados provenientes de várias fontes de informação, sentem a necessidade de encontrar técnicas que possibilitem reestruturar os conteúdos, sem que para isso se tenha que redefinir toda a organização como a informação que é gerada. Aqui levanta-se uma questão que é fundamental analisar: as fontes operacionais normalmente recebem tantas operações de adição e actualização de dados, que se torna prejudicial para o seu desempenho, efectuar grandes quantidades de consultas [WWW01].

As empresas e grupos económicos optam cada vez mais por implementar sistemas de *data warehousing* com o principal intuito de fazer o cruzamento da informação que é gerada nas fontes operacionais e corresponder directamente com as questões que os agentes de decisão definem

como sendo prioritárias. Obtém-se, assim, uma concreta correspondência entre os dados que são frequentemente gerados e as questões que os decisores consultam para fazer a gestão corrente dos seus negócios [WWW02]. Com a definição de esquemas dimensionais, organiza-se a informação de forma a possibilitar a consulta dos dados em diferentes perspectivas de análise e obter vários níveis de detalhe. Desta forma cria-se uma base em que facilmente se pode efectuar as consultas que mais frequentemente se elaboram para definir processos de negócio, mas também cria-se uma base onde se podem obter outros tipos de observações que as ferramentas de *reporting* tradicionais não permitem realizar, dado que são orientadas a um conjunto muito restrito de questões [Golfarelli et al. 2009].

Uma das formas mais utilizadas na exploração dos dados que se encontram armazenados em *data warehouses,* e que se observa um constante desenvolvimento nos últimos tempos, consiste na aplicação de ferramentas analíticas sobre este tipo de sistemas [WWW03]. Os agentes de decisão com uma ferramenta de processamento analítico de dados, devidamente implementada, têm à sua disposição uma plataforma aplicacional fortemente direccionada para as metodologias de trabalho utilizadas nas suas empresas. Desta forma, é possível aumentar a produtividade dos decisores e uma maior rapidez na satisfação das respostas a questões relativas a aspectos que giram em torno dos seus negócios.

Características como a rapidez da resposta às consultas, a flexibilidade da navegação sobre os dados contidos nas estruturas multidimensionais, a facilidade de mudança em tempo real dos diversos eixos de análise, providenciam uma maior eficiência na tomada de decisões e, consequentemente a produtividade dos decisores. São, pois, características como estas que revelam as enormes potencialidades das ferramentas de processamento analítico, tornando-as cada vez procuradas e utilizadas em meios empresariais [Golfarelli & Rizzi 2009].

Usualmente, os *data warehouses* guardam grandes quantidades de dados, provenientes de várias fontes de informação. Este grande volume de dados afecta, obviamente, a capacidade de resposta do sistema de *data warehousing* às consultas que lhe são lançadas pelos agentes de decisão. Isto faz com que, o processo de exploração de um *data warehouse* seja frequentemente muito dispendioso. De forma a atenuar este problema, as plataformas OLAP implementam, a par do *data warehouse*, vários serviços de processamento analítico alojados em servidores específicos capazes de responder de forma adequada às diversas pesquisas solicitadas pelas ferramentas de *front-end*. Na realidade, os tempos de resposta destes serviços são bons simplesmente porque muitos dos dados requeridos já estão materializados, de alguma forma, nessas plataformas no formato pretendido. Para isso é necessário alojar esses dados (no formato final pretendido ou num qualquer formato intermédio) *a priori* nesses servidores em estruturas específicas: os cubos. Para que tais estruturas fiquem disponíveis é necessário fazer o seu processamento que, em termos gerais, se resume fundamentalmente em três partes:

- **Computação do cubo** – Nesta fase os *data marts* definidos nos esquemas dimensionais vão ser analisados. As medidas presentes nas tabelas de facto e as hierarquias estabelecidas nos esquemas dimensionais, são convertidos em vistas multidimensionais (também

denominados por *cuboids*), organizados segundo todas as agregações possíveis. Portanto, nesta etapa calcula-se a *lattice* do cubo de dados.

- **Selecção das vistas multidimensionais** - A *lattice* do cubo usualmente representa um volume de dados enorme, muito maior do que aquele que efectivamente se encontra no *data warehouse*. Assim a materialização completa do cubo de dados torna-se impraticável, sendo apenas possível disponibilizar parte dos dados que constitui a *lattice* no servidor OLAP. No final do processo de selecção de vistas obtém-se o denominado cubo *iceberg*, que possui as vistas que efectivamente vão ser materializadas [Lawrence et al. 2006].

- **Carregamento do cubo de dados para o servidor OLAP** - Após efectuada a selecção dos conteúdos que poderão de facto ficar disponíveis no servidor OLAP, entra-se na última fase do processo de implementação de um cubo, que consiste, basicamente, no seu carregamento para o Servidor.

Na figura 1.1 está ilustrado o processo de implementação de um cubo de dados num Servidor OLAP. Nessa figura podem-se observar as duas plataformas de armazenamento de dados e as três fases que o processo de implementação de um cubo geralmente possui e apresentadas anteriormente.

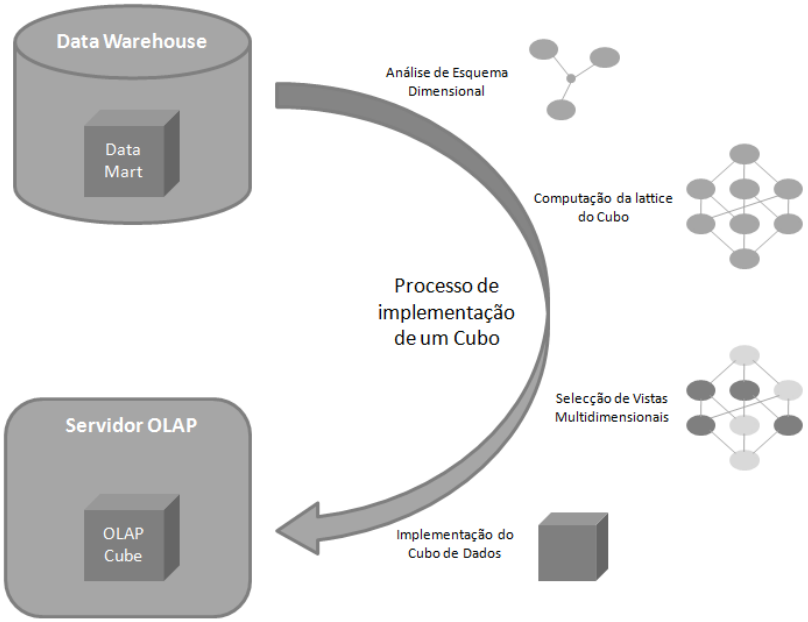

Figura 1.1 - Processo de implementação de um cubo de dados.

Uma das preocupações vulgarmente revelada pelos utilizadores de ferramentas analíticas é a de garantir as condições necessárias para que se consiga obter o máximo desempenho da ferramenta

e consequentemente tirar o máximo partido deste tipo de sistemas. Tudo isto porque é comum verificarem-se tempos de espera bastante prolongados para se obter uma qualquer resposta às consultas efectuadas pelos utilizadores. Assim, devem-se encetar esforços válidos no sentido de se melhorar a capacidade de resposta deste tipo de sistemas. Na realidade, na última década, muito se tem feito neste sentido. Basta ver na literatura da especialidade os inúmeros trabalhos realizados pelos investigadores e técnicos do domínio.

A selecção de vistas multidimensionais, realizada num processo de implementação de um cubo de dados, é uma tarefa que tem grande importância. Uma das formas de efectuar essa selecção, consiste em desenvolver mecanismos que identifiquem os conteúdos mais frequentemente consultados e corresponde-los com as vistas multidimensionais que fazem parte da *lattice* de um cubo. Para que este mecanismo de selecção se torne realmente efectivo, é necessário efectuar uma constante monitorização das consultas e, assim, realizar a selecção de vistas baseadas em sessões de exploração sobre o cubo de dados.

1.2 Selecção de Vistas Multidimensionais

A selecção do conjunto de vistas multidimensionais que forma um dado cubo *iceberg*, quando efectuada de forma despreocupada ou de forma superficial, pode incorrer num agravamento dos tempos de resposta das consultas executadas nas ferramentas analíticas de dados. Se o servidor OLAP em causa não possuir as capacidades mínimas para armazenar uma parte considerável da estrutura correspondente ao esquema dimensional envolvido, verificar-se-á que este servidor recorrerá de forma sistemática ao seu *data warehouse* de suporte para poder satisfazer a pesquisa requisitada por um utilizador.

Os algoritmos que propõem um determinado conjunto de vistas a materializar pretendem, sobretudo, encontrar um equilíbrio entre os recursos disponibilizados num servidor OLAP e os tempos de resposta das consultas realizadas [Chaudhuri et al. 1997]. Embora este propósito seja fácil de entender, encontrar uma solução que se revele óptima para todos os tipos de situações não é simples, dado que este equilíbrio de recursos tem que ser convertido e delineado em processos computacionais que sejam possíveis de implementar. Várias propostas surgiram para a resolução deste problema. Uma das primeiras em que de facto se procedeu a uma solução para seleccionar em subconjunto de vistas a ser indicado para materialização foi em apresentada em 1996 através da apresentação do algoritmo Greedy [Harinarayan et al. 1996]. Desde então, várias outras alternativas, mais evoluídas, têm sido avançadas, sendo que ainda hoje, com os desenvolvimentos que se verificam sobre as ferramentas analíticas de dados, se efectua a apresentação de novas propostas de resolução para este problema. Vemos, assim, que o tema ainda é bastante actual.

Um método que pode ser definido com o objectivo de tratar a selecção de vistas multidimensionais na fase da implementação do cubo, de forma a encontrar as vistas que devem ser efectivamente materializadas, poderia ser desenvolvido em torno da identificação de quais as vistas que mais são utilizadas pelos agentes de decisão durante as sessões de consulta que

desenvolvem. Uma forma de tornar isso possível, seria através da elaboração de processos capazes de supervisionarem os níveis de utilização de um cubo, podendo identificar quais as operações mais executadas, de acordo com o contexto em que os agentes de decisão estão inseridos. Uma vez identificadas essas vistas, da próxima vez que se proceder à actualização do cubo, ir-se-ão materializar apenas na sua estrutura os dados correspondentes a essas mesmas vistas.

Para que se consiga obter com um elevado grau de precisão as vistas mais consultadas durante as sessões de exploração sobre o cubo, vários parâmetros terão que ser analisados e vários níveis de destaque terão que ser atribuídos. Não basta, pois, identificar as dimensões de um cubo que são mais utilizadas, também se terá que identificar até que nível de detalhe elas são pesquisadas e que dados são consultados dentro de uma dimensão. Também se deverá fazer um controlo de forma a desenvolver indicações a uma escala temporal, para que se assim haja a possibilidade de seleccionar a informação dividida por intervalos de tempo. Só assim se poderá obter os dados que foram recentemente consultados, com um bom grau de precisão. Pretende-se desta forma evitar situações de erro e dar uma maior ênfase às vistas caso se verifique uma enorme quebra na sua utilização. Com o objectivo de não ultrapassar o limite de recursos disponíveis no servidor OLAP, deve-se ter em conta, também, o limite máximo de memória que um dado conjunto de vistas seleccionado poderá atingir. Várias são, portanto, as vantagens e as aplicações que se podem retirar com a devida combinação de processos que abordam as questões identificadas.

Depois de estudar os métodos de selecção avançados até ao momento, percebe-se que existe uma variedade muito grande de soluções que resolvem o problema de seleccionar um conjunto de vistas multidimenionais, no processo de implementação de um cubo [Qiu e tal. 2000]. O conjunto de abordagens e soluções apresentadas até hoje, são reflexo do esforço efectuado por parte dos investigadores da área dos Sistemas de Suporte à Decisão.

1.3 Motivação e Objectivos

A principal motivação que levou à realização dos trabalhos desta dissertação, consistiu em encarar o problema de selecção de vistas multidimensionais com o âmbito de definir uma alternativa válida aos métodos de selecção avançados até ao momento. Desenvolvendo processos que resultem na escolha de um conjunto de vistas no processo de selecção, na fase de implementação de um cubo OLAP, pretendeu-se definir um método que encontra o devido equilíbrio entre os recursos disponíveis e os desempenhos aceitáveis por partes da ferramentas analíticas de dados, independentemente do contexto e do tipo de esquema dimensional que se pretende implementar num servidor OLAP. A principal característica, que se pretende observar no conjunto de vistas seleccionado, é o de verificar uma forte probabilidade de ser consultado durante as futuras sessões OLAP. *A priori*, ao ser seleccionado o conjunto de vistas mais consultado, obtém-se uma diminuição do número de consultas sobre o *data warehouse*. Este é o principal facto que se pretende ver comprovado na fase de testes e análise de desempenho do método de selecção desenvolvido nesta dissertação.

Basicamente, a ideia que se pretende transmitir, é a de que existem vantagens em efectuar uma supervisão sobre os conteúdos consultados. Ao ser implementado este controlo sobre as consultas, poder-se-á obter um melhor entendimento sobre as questões fulcrais que constituem um negócio. Desta maneira obtém-se uma maior noção sobre a correspondência que se pretende alcançar entre os dados que são gerados e as decisões tomadas pelos agentes.

Os esforços realizados ao longo dos trabalhos desta dissertação consistiram essencialmente na definição de processos que culminaram na elaboração de um método de selecção, em que a base de funcionamento parte da análise sobre os níveis de utilização de um cubo. De forma mais específica, os objectivos que se definiram para este trabalho de dissertação foram os seguintes:

- Compreender a correspondência entre o comportamento de um utilizador durante os seus processos de exploração de cubos e os conteúdos que possivelmente consultará no futuro.
- Identificar os algoritmos de selecção existentes, de forma a formular ideias que adicionem progressos no desenvolvimento do método de selecção desenvolvido.
- Caracterizar os vários tipos de exploração realizados como forma de caracterizar um padrão de exploração OLAP.
- Desenvolver alguns processos e métricas precisas, a fim de perceber o nível de utilização de um cubo.

Um dos principais problemas a resolver quando se pretende identificar o conjunto de vistas mais frequentemente utilizado, é de facto o de atribuir a cada vista multidimensional um nível de utilização, para que assim se possa definir se é ou não vantajoso fazer a materialização da vista em questão. Outro problema que se pode constatar, é o de definir parâmetros de entrada que expressem restrições sobre o tipo de informação que se pretende seleccionar e da mesma forma, construir os cubos *iceberg* com fortes possibilidades de consulta. Todas estas questões, que compõem o novo método de selecção desenvolvido estão largamente discutidas ao longo da dissertação.

O método de selecção desenvolvido nos trabalhos desta dissertação teve como objectivo principal seleccionar as vistas mais consultadas de um dado cubo para que posteriormente se possa obter melhores desempenhos nas ferramentas de análise. Para além desta questão, que o algoritmo de selecção irá resolver, existem outras funcionalidades e indicações que poderão resultar com a utilização deste método, tão (ou mais) importantes do que o seu propósito principal. A identificação de partes do cubo que não são consultadas (o que significa que manter a respectiva informação no *data warehouse* é identificada como sendo desnecessária) é uma aplicação muito útil e que tem por base o mesmo mecanismo de análise sobre as *queries* de consulta. Os mecanismos de controlo sobre os dados consultados, implementados com neste método de selecção, poderão também ser utilizados para traçar perfis de trabalho e, desta forma, perceber a informação que os agentes de decisão focam a sua atenção para tomar as suas opções de negócio.

1.4 Estrutura da Dissertação

Além deste capítulo introdutório, este documento encontra-se estruturado em quatro outros capítulos que, genericamente, descrevem os estudos e desenvolvimentos efectuados no âmbito deste trabalho de dissertação. Esses capítulos são os seguintes:

- **Algoritmos de Selecção OLAP** - Neste capítulo, realizou-se uma análise sistemática sobre o estado da arte da tecnologia OLAP e sobre os principais algoritmos de selecção de estruturas multidimensionais de dados.
- **Selecção de Hiper-Cubos com Base em Padrões de Exploração OLAP** – Aqui, apresentamos de forma detalhada o método de selecção elaborado, tendo em conta os objectivos inicialmente propostos, descrevendo-se também a forma como se tirou partido da monitorização das sessões OLAP para obter a devida correspondência das vistas multidimensionais mais frequentemente consultadas.
- **Testes de Desempenho** - Neste capítulo apresentam-se os diversos testes de desempenho efectuados sobre o método de selecção desenvolvido, avaliando-se o seu comportamento comparando-o com outros algoritmos de selecção.
- **Conclusões e Trabalho Futuro** - Neste capítulo evidenciam-se as principais conclusões retiradas da realização deste trabalho, assim como se enunciam algumas linhas de orientação para a evolução do projecto.

Capítulo 2

Algoritmos de Selecção OLAP

No actual contexto do mercado das tecnologias da informação, que se encontra cada vez mais competitivo e difícil para conquistar posições de relevo por consideráveis períodos de tempo, as empresas enfrentam a necessidade de gerar vários tipos de informação associados aos seus métodos de negócio e seus intervenientes. A partir do momento em que os dados relativos aos processos de trabalho têm como origem diversos sistemas, que não se encontram baseados numa única fonte de informação, cria-se a necessidade de reunir todos esses dados e perceber a correspondência que possam ter com as diversas situações que os agentes de decisão estão preocupados em analisar, para que, assim, se possa efectuar um controlo mais efectivo do negócio em questão [Connolly & Begg 2001].

Uma aposta que os grupos e empresas têm vindo a investir nos últimos anos é na implementação de *data warehouses* como um meio privilegiado para centralizar a informação gerada pelos seus diversos pontos de negócio e para implementar ferramentas analíticas para a exploração dos seus dados [Chaudhuri & Dayal 1997]. Assim, os agentes de decisão têm a seu dispor uma forma fácil e intuitiva de consultar os factos relativos aos seus negócios, podendo obter as respostas para as questões que constantemente precisam ver satisfeitas [Kimball et al. 2008].

Neste capítulo, irá abordar-se a tecnologia OLAP, tentando-se apresentar uma definição concreta associada ao tipo de sistemas referido e explicar as vantagens que advêm com a sua utilização. Após feita esta introdução, irá abordar-se os seus processos mais comuns de acesso à informação para que os agentes de decisão consigam obter as respostas que procuram nas várias análises que desenvolvem sobre o seu sistema de informação. De seguida, vão ser apresentados os principais tipos de sistemas OLAP hoje em dia disponíveis, explicando-se as suas principais características. Por fim, serão abordados alguns algoritmos de selecção OLAP, explicando a sua forma de actuação, bem com os critérios e processos que executam, para que, assim, seja realizado um

processo de revisão sobre as principais questões que os algoritmos de selecção existentes usualmente levantam.

2.1 Processamento Analítico de Dados

Hoje em dia, os grupos e empresas que ocupam posições competitivas no mercado encontram-se numa situação em que necessitam cada vez mais de armazenar informação relacionada com os recursos utilizados na gestão dos seus negócios, sejam eles humanos, materiais ou financeiros. Essa informação, que é cada vez mais volumosa e descentralizada, torna-se indispensável para que os agentes de decisão efectuem as suas tarefas quotidianas e obtenham uma boa gestão dos seus negócios [Connolly & Begg 2001]. Outra necessidade, revelada por grupos económicos e empresas, e que se tem vindo a se desenvolver ao longo destes últimos anos: o cruzamento de toda a informação armazenada, não apenas de uma forma periódica e talhada para responder a um conjunto estático de questões, mas de forma a poder efectuar um controlo constante e vocacionado a uma sessão de trabalho específica, em que várias comparações e análises podem ser levadas a cabo. Ao efectuar estas relações, podem-se obter observações muito conclusivas para quem tem que analisar o negócio como um todo [Kimball et al. 2008].

Com vista à obtenção de um ambiente que permita encontrar soluções para dar resposta, por um lado, à necessidade de se obter uma forma rápida para fazer o cruzamento de dados existentes em vários sistemas operacionais, e por outro, para efectuar pesquisas em que as questões colocadas de forma *ad hoc* têm espaço e tempo para serem efectuadas [Golfarelli et al. 2009], hoje em dia são cada vez mais as empresas que enveredam pela implementação de *data warehouses*, bem como pela aplicação de ferramentas de processamento analítico de dados. De facto, com a utilização desta combinação de conceitos e práticas, muitas são as vantagens que se podem destacar - a disponibilização de um sistema muito flexível para análise de dados é, por exemplo, um dos muitos aspectos fulcrais para quem está a gerir um negócio em mercados tão competitivos como os que se observam hoje em dia [Chaudhuri & Dayal 1997].

Na literatura da área, OLAP (Online Analytical Processing) é definido como sendo o termo que descreve a tecnologia em que se efectua a síntese, análise e a consolidação de grandes volumes de dados organizados em estruturas multidimensionais, de forma dinâmica e interactiva, em que se obtêm ambientes de acesso rápido para efectuar análises avançadas sobre informação estrategicamente armazenada [Golfarelli & Rizzi 2009][Connolly & Begg 2001]. As ferramentas OLAP são as mais indicadas, ou pelo menos as mais utilizadas, para se proceder à exploração de informação sobre um *data warehouse*, dado que oferecem aos seus utilizadores a oportunidade de analisar e explorar os dados interactivamente sobre um dado modelo multidimensional, durante as suas várias sessões de consulta.

2.1.1 Vantagens das Aplicações OLAP

As vantagens que potencialmente se poderão tirar partido, após uma implementação bem sucedida de uma aplicação OLAP, são [Golfarelli & Rizzi 2009]:

- O aumento de produtividade dos agentes de decisão e, consequentemente, de toda a organização, uma vez que um acesso mais fácil e rápido à informação pode permitir uma maior eficiência dos processos de tomada de decisão.
- O aumento potencial da receita e dos lucros, dado que a aplicação OLAP permite a uma organização responder mais rapidamente às exigências do mercado.
- A menor dependência dos agentes de decisão das equipas de desenvolvimento e de manutenção. As aplicações OLAP proporcionam ambientes de consulta de fácil utilização e de interpretação de resultados em ambientes de fácil aprendizagem e utilização.
- A informação apresentada pelas aplicações OLAP é credível e devidamente controlada, dado que os dados têm como proveniência, usualmente, um sistema de *data warehousing*.
- A redução do tráfego de rede e de consulta sobre os dados armazenados num *data warehouse*, já que as aplicações carregam para o servidor OLAP grande parte dos dados a disponibilizar aos seus utilizadores finais.

2.1.2 Componentes Gerais de um Sistema OLAP

Após todos os sistemas estarem devidamente implementados e configurados, os registos que se obtêm a partir dos sistemas operacionais irão passar, de forma genérica e dependendo da categoria de sistema OLAP implementado, por mais duas componentes principais de um sistema de data warehousing, nomeadamente o *data warehouse* e o servidor OLAP. Será neste último servidor que a informação ficará em condições para ser trabalhada pelas aplicações de *front end*, que por sua vez proporcionará aos utilizadores finais as operações e os métodos de análise usualmente disponíveis em ambientes OLAP [Cuzzocrea et al. 2009]. Na figura 2.1 [Chaudhuri et al. 2001] estão representados todos os componentes e processos necessários para que as aplicações de *front end* possam aceder à informação actualizada e estrategicamente organizada. Os componentes referidos são os seguintes:

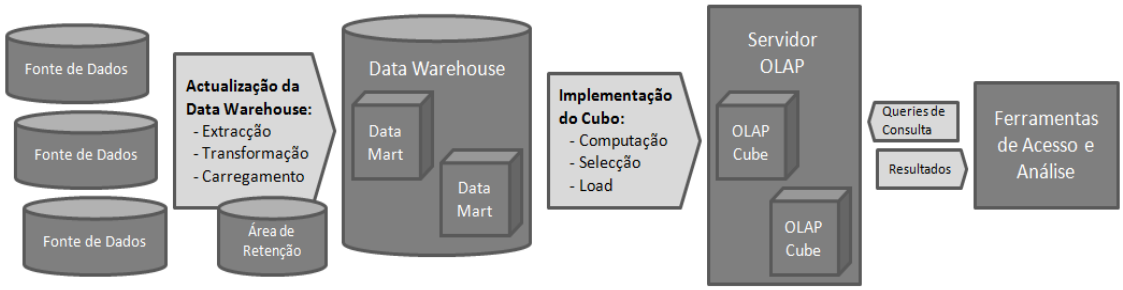

Figura 2.1 - Componentes gerais de um Sistema OLAP.

- **Fontes de Dados** – São nas fontes de dados que os registos são adicionados, actualizados e removidos através das aplicações operacionais. Já neste componente se utilizam alguns métodos para auxiliar o processo de actualização do *data warehouse*, com a implementação de tabelas de auditoria e variáveis de controlo sobre os registos, entre outros.

- **ETL** - O processo de actualização de um *data warehouse* pode ser dividido em três passos essenciais: o processo de extracção dos dados a partir dos sistemas operacionais, o processo de transformação dos dados extraídos em que se efectuam operações de tratamento para obter a consolidação de informação a actualizar e, por fim, o processo de carregamento para o *data warehouse*. Para suportar todos os processos de ETL é usual implementar-se uma área de retenção de dados (*staging area*) que serve de suporte às operações que são necessárias realizar sobre os dados extraídos e para implementar mecanismos que garantam, por exemplo, a recuperação do sistema em caso de ocorrência de uma situação de erro.

- **Data Warehouse** - No *data warehouse* estão contidos os dados de forma consolidada e organizada segundo um esquema multidimensional. A informação obtida nas fontes de dados é convertida para o conjunto de questões que os agentes de decisão necessitam ver constantemente supervisionados. Esta separação é fundamental, dado que é a forma mais eficiente de se criar um ambiente de consulta sobre a informação, que normalmente consome um grande volume de dados e assim as *queries* de consulta consomem muitos recursos.

- **Implementação do Cubo** - Independentemente do método de implementação utilizado, o processo de implementação de um cubo pode ser dividido em três processos fundamentais. De grosso modo, podemos dividi-los em métodos ROLAP (Relational-OLAP), MOLAP (Multidimensional-OLAP) e HOLAP (Hybrid-OLAP). Cada um destes processos, independentemente do seu tipo, executa sempre a computação dos cubos e a selecção das vistas a serem materializadas e carregadas para o servidor OLAP.

- **Servidor OLAP** – É no servidor OLAP que estão colocados os cubos correspondentes aos *data marts* implementados no *data warehouse*. É sobre estas estruturas de dados que os agentes de decisão lançam as suas *queries* que o servidor OLAP responderá, enviando às aplicações de *front-end* os seus resultados.

- **Ferramentas de Acesso e Análise** - As ferramentas de *front-end* possibilitam aos utilizadores finais efectuar as suas consultas sobre um ou mais cubos, de forma intuitiva e simples, e fazer a análise dos resultados obtidos. Com estas ferramentas os agentes de decisão terão acesso a uma plataforma de análise bastante sofisticada, usualmente reparada de acordo com os requisitos de análise estabelecidos para o seu negócio.

2.1.3 Características de um Sistema OLAP

Em 1970, Edgar Frank Codd escreveu um artigo [E. F. Codd 1970] em que definiu doze regras - as designadas 12 regras de Codd - com o propósito de demonstrar os fundamentos teóricos dos

sistemas de gestão de base de dados relacionais. Graças a essas doze definições houve um enorme avanço e desenvolvimento na organização de dados em grande escala, contribuindo de forma decisiva para a evolução dos motores de base de dados. Mais tarde, em 1993, Codd e mais dois outros colegas publicaram um novo artigo [Codd et al. 1993] em que tentaram definir o termo OLAP baseando-se nessas doze regras, tentando estabelecer as propriedades que as aplicações baseadas na tecnologia OLAP deveriam disponibilizar aos seus utilizadores. Embora estas doze regras tenham sido aceites na generalidade pela comunidade científica, outras linhas de trabalho desenvolvidas por outros investigadores e especialistas do domínio tentam completar e adicionar mais características e restrições a esse conjunto inicial de doze regras. Por exemplo, o Gartner Group adicionou mais nove regras e, mais tarde, a IRI Software propôs mais três novas definições [Koutsoukis et al. 1999]. Todavia, as doze regras apresentadas por Codd são ainda encaradas como uma introdução e uma forma de avaliação dos sistemas OLAP.

De seguida, apresentam-se as doze regras que Codd introduziu, de forma a apresentar as características e funcionalidades gerais que uma ferramenta ou aplicação OLAP deve possuir [Codd et al. 1993]:

- **Conceito de vistas multidimensionais** - Os agentes de decisão devem aceder aos dados relativos aos seus negócios numa perspectiva organizada de forma multidimensional, assim as vistas que suportam os dados devem possuir operações de *roll-up*, *drill-down*, *slice* e *dice*.
- **Transparência** - Para assegurar que num sistema OLAP se possa facilmente acrescentar novas operações e funcionalidades, ou então ser adicionado a um sistema já em funcionamento, as aplicações OLAP devem ser transparentes. Deve haver a preocupação de desenhar uma arquitectura aberta, de forma a poderem ser embutidas noutros sistemas sem apresentar impactos adversos.
- **Acessibilidade** - As ferramentas OLAP devem aceder e executar um correcto mapeamento entre o seu próprio esquema lógico e qualquer outro tipo de base de dados heterogénea, aceder aos dados sem introduzir restrições de leitura e possibilitar possíveis conversões que se tornem necessárias, de forma a elaborar um simples e coerente acesso aos dados.
- **Performance consistente na obtenção de relatórios** - Conforme a dimensão dos esquemas dimensionais aumentem ou o volume de dados aumente, os sistemas OLAP não devem mostrar sinais de degradação ao longo do funcionamento do sistema. Garante-se desta forma, que os utilizadores finais possam efectuar as suas sessões de consulta sem se preocuparem em resolver problemas relacionados com os desempenhos dos seus sistemas.
- **Arquitectura cliente/servidor** - Os dados dos sistemas OLAP são armazenados em servidores centrais, desta forma é essencial que se projectem arquitecturas cliente/servidor. Os servidores devem ser pensados de forma a permitirem a conexão de múltiplos clientes sem que haja uma degradação acentuada sobre o servidor central.
- **Dimensionamento genérico** - Todas as dimensões presentes no esquema dimensional devem receber igual tratamento relativamente ao conjunto de operações possíveis de se efectuar. Assim garante-se que todas as operações sugeridas pela tecnologia OLAP possam ser executadas em qualquer uma das dimensões. Sempre que se acrescentem novas

funcionalidades numa ferramenta OLAP, essa mesma funcionalidade deve ser acessível em todo o esquema dimensional.

- **Tratamento dinâmico de matrizes esparsas** - Os esquemas físicos das aplicações OLAP devem-se adaptar à dispersão (percentagem de valores em falta de acordo com os registos armazenados) que os valores dos registos assumem, mas também devem-se adaptar à forma como os dados se encontram distribuídos.
- **Suporte a multi-utilizadores** - As ferramentas OLAP devem suportar e resolver o acesso e manipulação concorrente aos dados presentes num servidor OLAP.
- **Operações de cruzamento de dimensões** - As agregações que são permitidas dentro do esquema dimensional devem ser inferidas e materializadas de forma a evitar os cálculos repetitivos durante a consulta sobre os valores armazenados no servidor. Desta forma as ferramentas OLAP não podem introduzir restrições no cálculo entre dimensões presentes no esquema dimensional.
- **Intuitiva manipulação dos dados** - Operações de manipulação sobre os dados deve ser efectuada directamente sobre o modelo analítico e desta forma, as estruturas que suportam as vistas presentes no modelo devem possuir as operações de *drill across*, *zoom out*, entre outras funcionalidades de navegação sobre os dados.
- **Relatórios flexíveis** - Os ambientes de análise e apresentação dos dados devem ser orientados da forma mais simples possível, ou seja por linhas, colunas e células de dados que podem sem visualmente comparados de forma intuitiva. Os relatórios devem possuir funcionalidades de apresentação de informação sintetizada, proporcionando uma organização esclarecedora e que torne as operações sugeridas pela tecnologia OLAP de fácil execução para um determinado utilizador.
- **Ilimitados níveis de dimensões e agregações** - Durante as sessões de consulta, as ferramentas OLAP não podem introduzir restrições quanto aos níveis de operações sucessivas dentro de uma dimensão, de forma a proporcionar operações de uma forma contínua durante a navegação sobre um cubo.

2.2 Tecnologia OLAP

As ferramentas OLAP, devido às suas propriedades, tornam-se nos sistemas mais acessíveis para efectuar consultas e análises sobre os dados presentes num *data warehouse*, dado que são ferramentas pensadas e organizadas para tirar partido das características dos modelos dimensionais [WWW08]. Desta forma, os utilizadores deste tipo de tecnologia possuem práticas avançadas para efectuar a gestão dos seus negócios, sempre com a garantia que aspectos como a performance e bom funcionamento são tidos sempre em conta. Muitos dos processos executados

são efectuados com o intuito de satisfazer uma das suas prioridades principais: a obtenção de curtos tempos de espera durante as consultas sobre um cubo, presente no servidor OLAP.

Se a aplicação da tecnologia OLAP for efectuada com sucesso sobre um *data warehouse*, um agente de decisão encontra-se na possibilidade de obter facilmente respostas às questões que necessita ver regularmente supervisionadas, uma vez que o modelo analítico proporciona uma visão multidimensional sobre os dados. O facto de estarem elaboradas tabelas factos com medidas indicativas sobre cada registo e com dimensões organizadas segundo hierarquias, desenvolvem-se propriedades únicas que dificilmente podem ser combinadas noutro tipo de sistemas. A fácil consulta sobre dados históricos, consultas automáticas sobre os dados, a possibilidade de observar os dados segundo diferentes perspectivas, efectuar operações e cálculos entre diferentes perspectivas de análise, são exemplos de operações que neste tipo de sistemas existe a possibilidade de implementar e dificilmente podem ser combinadas da mesma forma noutro tipo de organização de dados.

Em comparação com as ferramentas de *reporting* convencionais (em que não existe qualquer tipo de flexibilidade de consulta sobre os dados) as ferramentas OLAP surgem cada vez mais como uma alternativa fiável para cobrir as necessidades, que surgem sobre os agentes de decisão quando possuem apenas relatórios orientados a um conjunto estático de questões [WWW09]. A versatilidade nas sessões consultas, permitem a um decisor possuir uma visão mais flexível e por vezes mais ampla sobre o negócio que se encontra a gerir.

Este tipo de sistemas pode assumir principalmente três tipos de configurações, os sistemas ROLAP, MOLAP e HOLAP. Fica ao critério de cada utilizador perceber as características que cada arquitectura possui e identificar a que mais se adequa conforme as suas necessidades e disponibilidade de recursos.

2.2.1 Cubo de Dados

De forma geral pode-se dizer que a modelação dimensional é uma técnica em que se elaboram modelos lógicos (chamados de esquemas dimensionais) que organizar os dados, a serem armazenados fisicamente numa base de dados, com o intuito principal de introduzir performance no acesso aos dados [R. Kimball 1997].

Todos os esquemas dimensionais são compostos por tabelas de facto, que são tabelas que possuem uma chave primária que é definida através de uma composição de chaves estrangeiras (que referenciam tabelas onde são expressas dimensões). Cada entrada numa tabela de facto possui medidas associadas (que normalmente assumem valores do tipo numérico), em que desta forma a ocorrência de uma combinação de valores referenciados pelas chaves estrangeiras (com valores associados às medidas) são chamados de factos. Nas tabelas que suportam as dimensões são definidos atributos que contêm informação descritiva sobre cada entrada que é utilizada na tabela de factos.

Na Figura 2.2 está representado um exemplo de um esquema dimensional em que se define uma tabela de factos em que o grão corresponde ao lucro obtido numa determinada venda, de um determinado produto, numa determinada data.

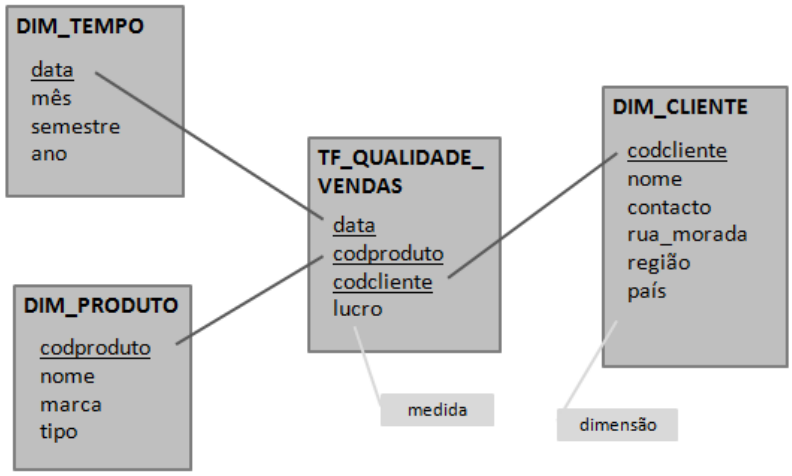

Figura 2.2 - Exemplo de um esquema dimensional.

Um cubo de dados [Chaudhuri & Dayal 1997] é usado para representar as vistas de dados que se encontram num esquema dimensional [Sapia el al. 1998], agrupadas pelas possíveis agregações de que se pode efectuar e consoante as hierarquias estipuladas dentro das dimensões. Na figura 2.3 está representado visualmente um cubo de dados correspondente ao esquema apresentado na figura 2.2. Cada célula que compõe o cubo corresponde a um facto presente na tabela de factos e o conteúdo dentro da célula do cubo representam as medidas do facto correspondente, por fim os eixos do cubo representam as dimensões, ou as perspectivas de análise sobre os factos que estão armazenados no cubo.

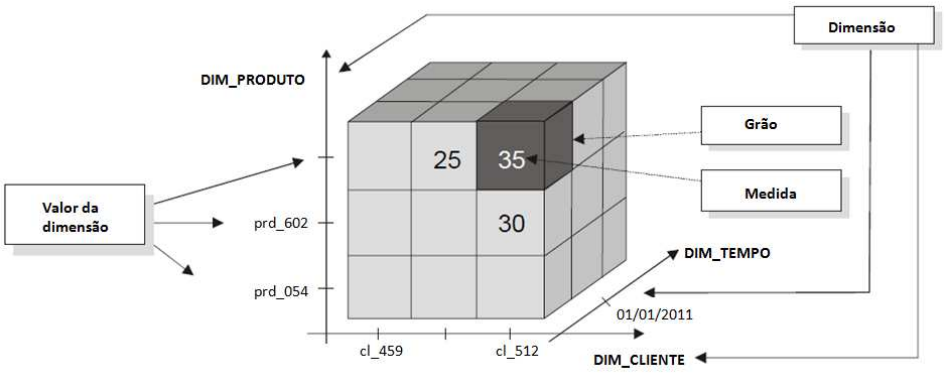

Figura 2.3 - Cubo correspondente ao esquema dimensional apresentado na figura 2.2.

A passagem dos dados presentes num esquema dimensional (fisicamente presentes num *data warehouse*) para um cubo de dados a ser materializado num servidor OLAP é essencial, dado que são estruturas que fazem a correcta correspondência entre os dados armazenados e as

funcionalidades que a tecnologia OLAP fundamenta, mas também dado que o cubo possui pré-calculadas e armazenadas as vistas correspondentes ao esquema dimensional, assim não é necessário efectuar repetidamente as associações entre as dimensões do esquema sempre que pretendido, resultando numa maior performance no momento das consultas sobre os dados.

2.2.2 Sessões de Consulta

Uma sessão OLAP consiste na navegação e exploração de um cubo, que resulta da aplicação de sucessivas operações que correspondem ao processo de analisar os factos armazenados no cubo, com diferentes perspectivas e com diferentes níveis de detalhe. As ferramentas OLAP habitualmente apresentam os dados obtidos através das *queries* de consulta na forma de tabelas com múltiplos cabeçalhos, cores e outras indicações e funcionalidades sugestivas para realçar os resultados [Kimball et al. 2008]. Os resultados das *queries* assumem estruturas multidimensionais e que portanto seria muito difícil para um ser humano proceder à sua interpretação, nos casos em que os resultados assumem mais do que três dimensões [Golfarelli et al. 2009]. Cada operação efectuada durante a sessão de exploração sobre o cubo é caracterizada por um operador OLAP, que reformula o estado de exposição do cubo. Os operadores mais comuns são o *roll-up*, *drill-down*, *slice-and-dice*, *pivot*, *drill-across* e *drill-through* [Golfarelli et al. 2009].

Roll-up

O operador *roll-up* reduz o nível de detalhe sobre os dados seleccionados, obtendo um resumo dos dados dentro da hierarquia. Nas figuras 2.4 e 2.5 [Cios et al. 2007] encontra-se exemplificado a aplicação do operador *roll-up*, em que o nível de detalhe da dimensão que correspondente aos meses correspondentes às datas em que se efectuaram as vendas, foi diminuído, tendo sido agrupados/agregados em trimestres.

# sold units		2002					
		January	February	March	April	May	June
CPU	Intel	442	224	211	254	187	112
	AMD	401	289	271	208	234	267

Figura 2.4 - Apresentação do cubo antes da aplicação do operador *roll-up*.

# sold units		2002	
		Quarter 1	Quarter 2
CPU	Intel	877	553
	AMD	961	709

Figura 2.5 - Apresentação do cubo após a aplicação do operador *roll-up*.

Drill-down

O operador *drill-down* aumenta o nível de detalhe sobre os dados seleccionados, obtendo uma análise mais esclarecedora dos valores dentro da hierarquia. Nas figuras 2.6 e 2.7 [Cios et al. 2007] encontra-se exemplificado a aplicação do operador *drill-down*, em que o nível de detalhe da dimensão que correspondente aos locais de vendas em foi aumentado, tendo sido detalhado pelas regiões presentes no sistema que compõe o país USA.

# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
All	USA	2231	2134	1801	1560	1129
	Europe	1981	2001	1432	1431	1876

Figura 2.6 - Apresentação do cubo antes da aplicação do operador *drill-down*.

# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
All	Denver	877	961	410	467	620
	LA	833	574	621	443	213
	NY	521	599	770	650	296

Figura 2.7 - Apresentação do cubo após a aplicação do operador *drill-down*.

Pivot

O operador *pivot* reformula a representação dos dados de acordo com uma nova perspectiva de análise, por outras palavras executa uma rotação no cubo sobre as dimensões seleccionadas. Na figura 2.8 [Cios et al. 2007] está representado a aplicação do operador *pivot* sobre um cubo de dados.

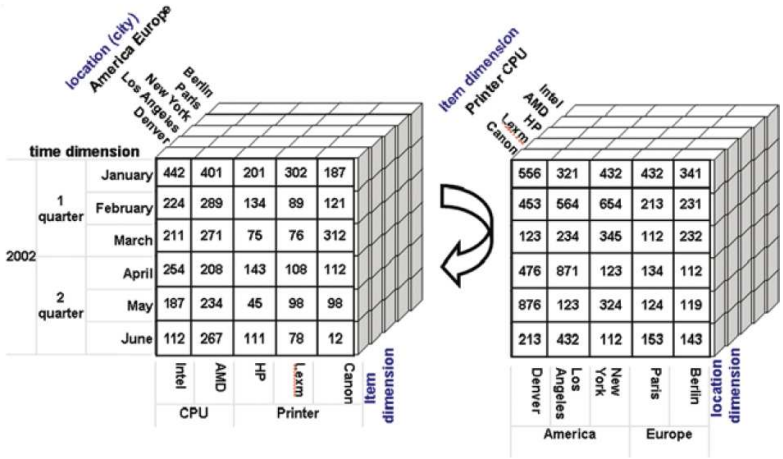

Figura 2.8 - Exemplo da aplicação do operado *pivot* sobre um cubo de dados.

Slice-and-dice

O operador *slice-and-dice* executa uma selecção de um subconjunto do cubo aplicando um determinado critério. Na figura 2.9 [Cios et al. 2007] está representado a aplicação do operador *slice-and-dice*, em que se observa o processo de obtenção uma célula do cubo.

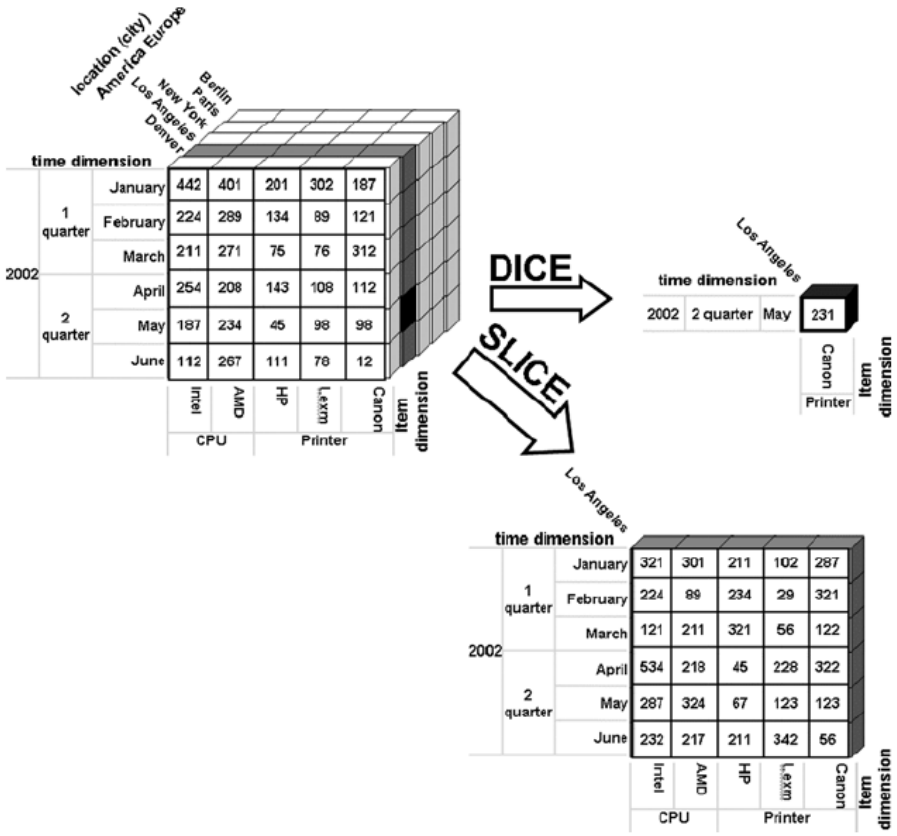

Figura 2.9 - Aplicação do operador *slice-and-dice*.

Drill-through

O operador *drill-through* permite efectuar a ligação entre os dados armazenados no cubo de dados presente no servidor OLAP e os correspondentes dados armazenados nos sistemas de origem, ou seja no *data warehouse*.

Drill-across

O operador *drill-across* estabelece a comparação entre cubos relacionados, comparando os seus dados, estabelecendo a correspondência entre os factos e as dimensões a serem representados. Na figura 2.10 e 2.11 [Golfarelli et al. 2009] representa-se a aplicação do operador *drill-across*, em

que se observa uma junção de valores de dois cubos. Obtém-se assim uma forma fácil de comparar valores de dois cubos logicamente relacionados.

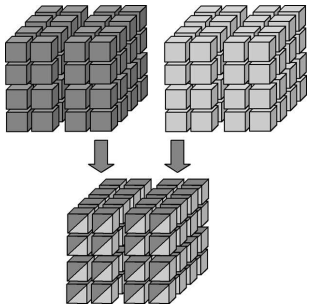

Figura 2.10 - Representação da junção de dois cubos.

Category	Metrics / Quarter	Revenue 2005 Q1	2005 Q2	2005 Q3	2005 Q4	2006 Q1	2006 Q2	2006 Q3	2006 Q4
Books		$319,767	$313,339	$336,862	$350,617	$348,483	$387,849	$407,392	$419,563
Electronics		$4,448,112	$4,299,411	$4,918,673	$5,633,676	$5,411,499	$5,714,783	$5,999,174	$6,528,576
Movies		$228,108	$232,201	$264,471	$307,611	$299,531	$326,270	$334,143	$373,182
Music		$168,843	$169,462	$193,234	$217,427	$212,438	$228,289	$239,112	$260,298

Category	Quarter / Metrics	2005 Q1 Discount	2005 Q1 Revenue	2005 Q2 Discount	2005 Q2 Revenue	2005 Q3 Discount	2005 Q3 Revenue	2005 Q4 Discount	2005 Q4 Revenue	2006 Q1 Discount	2006 Q1 Re
Books		$ 0	$319,767	$ 10,845	$313,339	$ 9,497	$336,862	$ 18,279	$350,617	$ 0	
Electronics		$ 0	$4,448,112	$ 150,366	$4,299,410	$ 143,395	$4,918,673	$ 302,884	$5,633,675	$ 0	$5
Movies		$ 0	$228,108	$ 8,025	$232,201	$ 7,948	$264,471	$ 16,649	$307,611	$ 0	
Music		$ 0	$168,843	$ 6,143	$169,462	$ 5,563	$193,234	$ 11,047	$217,427	$ 0	

Figura 2.11 - Exemplo da aplicação do operador *drill-across*.

2.2.3 Categoria de Sistemas OLAP

Os métodos de implementação de um cubo podem assumir várias vertentes e podem ser divididas em várias categorias. Cada categoria é classificada de acordo com a arquitectura das estruturas em que o cubo ficará implementado. Embora haja referências de outros métodos de implementação de cubos OLAP, as três principais categorias são os modelos ROLAP, MOLAP e HOLAP [Golfarelli & Rizzi 2009] [Connolly & Begg 2001].

Para além destes modelos existem referências aos métodos de implementação baseados em grafos [Lakshmanan et al. 2003] [Sismanis et al. 2002] em que usualmente se implementa o cubo em estruturas sob a forma de árvores e por fim os métodos baseados em aproximações [Vitter et al. 1999] [Gunopulos et al. 2000] que usam várias representações em memória, tal como histogramas. Nesta secção, que tem como objectivo principal o de falar sobre os algoritmos de selecção de vistas a materializar no processo de implementação de um cubo, não vamos dar ênfase aos métodos de implementação baseado em grafos nem nos métodos baseados em

aproximações, dado que estes dois modelos baseiam-se em implementações muitos específicas e não usam modelos de selecção genéricos de referência.

Sistemas ROLAP

Os servidores ROLAP (Relational OLAP) são constituídos por servidor *back-end* onde os dados estão armazenados num *data warehouse* (fisicamente implementado numa base de dados relacional) e pelas ferramentas *front-end* dos clientes. Desta forma, os servidores ROLAP exploram todas as vantagens que advêm com a utilização dos motores de bases de dados convencionais, tais como a escalabilidade no armazenamento de dados, fácil integração e ligação para diferentes sistemas, fácil acesso e selecção dos conteúdos guardados e as todas funcionalidades transaccionais das bases de dados relacionais. Os sistemas ROLAP suportam consultas multidimensionais e normalmente optimizadas especificamente para a sua utilização em bases de dados relacionais.

Na figura 2.12 [Connolly & Begg 2001] está representada uma arquitectura típica de um servidor ROLAP.

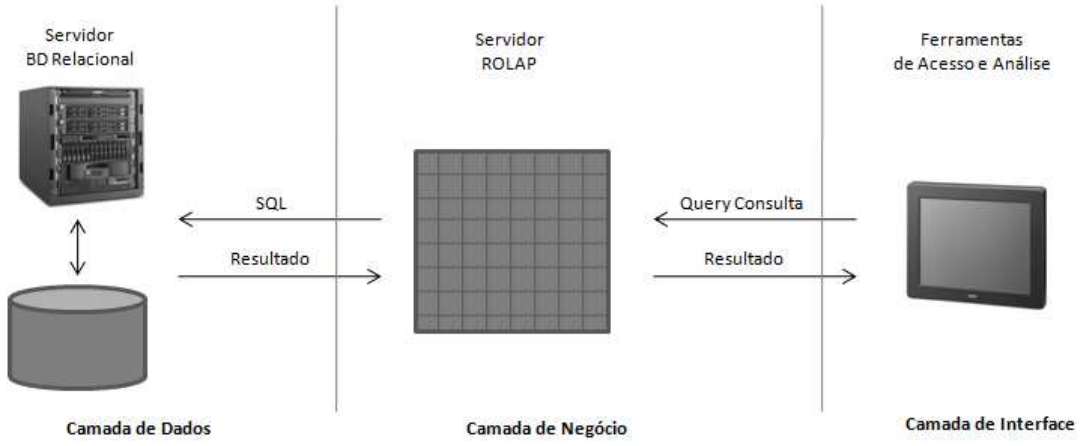

Figura 2.12 - Componentes gerais de um servidor ROLAP.

As principais desvantagens da utilização desta tecnologia são as seguintes:

- Exige a implementação de procedimentos que faça a conversão das estruturas com relações bidimensionais em estruturas multi-dimensionais;
- Os desempenhos das consultas efectuadas sobre os cubos implementados em servidores ROLAP apresentam piores valores do que os servidores MOLAP e HOLAP.

Sistemas MOLAP

A arquitectura MOLAP caracteriza-se principalmente pelo facto do *back-end* não ter como suporte uma base de dados relacional, em vez disso o suporte é feito através de *arrays* multidimensionais [Hasan et al. 2007]. A principal vantagem desta forma de organização é o suporte de vistas multidimensionais, permitindo a implementação de consultas com mapeamento directo sobre a camada de armazenamento. Estes sistemas têm associados propriedades de indexação próprias para esquemas multi-dimensionais e assim, o desempenho das *queries* de consulta possuem um desempenho consideravelmente elevado.

Na figura 2.13 [Connolly & Begg 2001] está representada uma arquitectura típica de um servidor MOLAP.

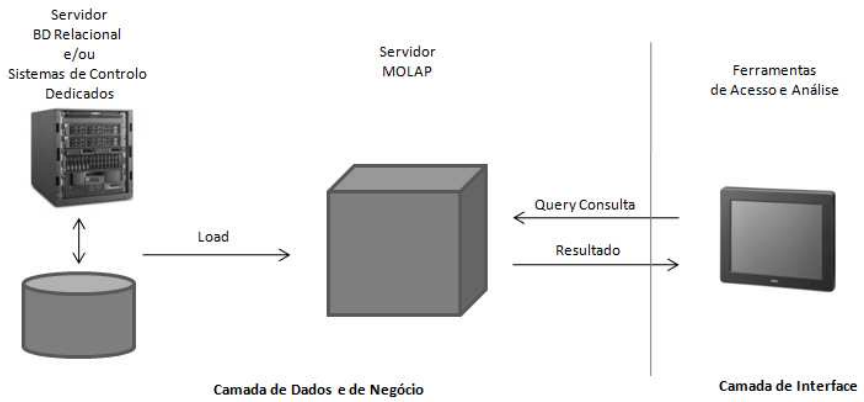

Figura 2.13 - Componentes gerais de um servidor MOLAP.

As principais desvantagens da utilização deste tipo de tecnologia são as seguintes:

- Para efectuar a manutenção dos conteúdos armazenados nos servidores MOLAP são necessários outros tipos de conhecimentos de análise e ferramentas específicas. Desta forma efectuar o seu suporte trás mais custos associados;
- As estruturas associadas a esta tecnologia são muito limitadas para suportar certos tipos de funcionalidades. Muitas ferramentas MOLAP fazem a gestão de informações específicas e com níveis de detalhe avançados em bases de dados relacionais implementadas separadamente;
- Embora os servidores MOLAP ofereçam um bom nível de desempenho, possui limitações quanto à capacidade de escalabilidade para enormes volumes de dados.

Sistemas HOLAP

A arquitectura HOLAP combina as tecnologias inerentes aos sistemas ROLAP e MOLAP. Desta forma possuem mecanismos para fazer frente aos comportamentos impróprios em termos de desempenho das arquitecturas MOLAP quando a quantidade de dados se torna muito grande e das arquitecturas ROLAP quando o volume de dados não atinge quantidades consideráveis. Efectuando a adequada utilização entre as abordagens ROLAP e MOLAP, dependendo da quantidade de recursos a serem utilizados e o formato dos esquemas multidimensionais dos diferentes cubos a armazenar.

Os processos de selecção de vistas a materializar na fase de implementação de um cubo, a indexação dos conteúdos armazenados e os processos de selecção sobre os cubos são similares aos utilizados nos sistemas ROLAP e MOLAP.

Na figura 2.14 [Connolly & Begg 2001] está representada uma arquitectura típica de um servidor HOLAP.

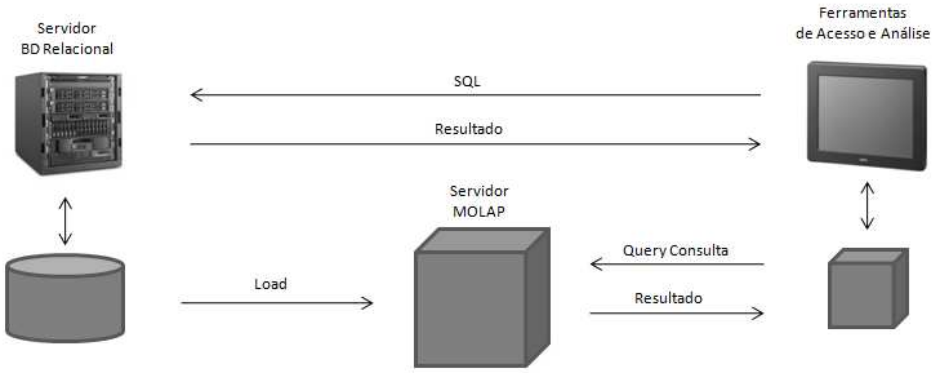

Figura 2.14 - Componentes gerais de um servidor HOLAP.

As principais desvantagens da utilização deste tipo de tecnologia são as seguintes:

- A utilização desta arquitectura pode resultar no surgimento de alguma redundância ao nível dos conteúdos armazenados, o que pode trazer problemas associados quando se verificam vários utilizadores a aceder aos dados geridos pelo servidor;
- Apenas um grupo limitado de informação poderá ser eficientemente controlado.

Escolha da arquitectura mais indicada

Cada uma das arquitecturas tem um contexto próprio. Uma das questões principais, é perceber quais as funcionalidades que a aplicação de *front-end* possui implementadas, desta forma existe a necessidade de identificar e escolher uma arquitectura que suporte as APIs utilizadas pelas ferramentas de *front-end* e identificar linguagens de *quiering* que o servidor deve suportar, para

que assim, se garanta o bom funcionamento das consultas enviadas para o servidor OLAP [C. Thomsen 2008]. Aspectos como os preços de implementação do sistema, recursos físicos necessários face às necessidades identificadas, manutenção do sistema após estar em funcionamento poderão ser decisivos, dado que o preço final da solução, dita em grande parte das vezes a sua aprovação.

Os sistemas que se apresentam como sendo os mais estáveis são os que são utilizam uma arquitectura ROLAP, dado que assenta os modelos dimensionais em sistemas de gestão de base de dados convencionais. Contudo, deve-se optar por uma arquitectura HOLAP quando o esquema dimensional se torna muito grande e com uma quantidade de dados armazenados considerável. Neste caso o servidor HOLAP contem as melhores soluções, dado que implementa vários mecanismos que visam desenvolver uma maior rapidez no acesso aos dados contidos no servidor. De facto a implementação de sistemas MOLAP não são tão eficientes, quanto os vendedores deste tipo de sistemas alegam - quando o *data warehouse* encontra-se fisicamente implementada num sistema de gestão de base de dados relacional. Os sistemas MOLAP assenta-se em implementações pouco genéricas e para se tirar partido das suas vantagens aconselha-se que o próprio modelo relacional já possua implementações de estruturas que efectuem ligações de acesso aos dados sobre o esquema dimensional.

2.3 Algoritmos de Selecção OLAP

A fase de implementação de um cubo é de extrema importância, dado que grande parte do desempenho das sessões de consulta depende dos métodos utilizados nesta etapa, que necessitam de ser os mais apropriados, para que no fim se obtenha o devido equilíbrio entre os recursos disponíveis e os tempos de resposta das *queries* de consulta efectuadas sobre o cubo [Harinarayan et al. 1996].

Independentemente do método de implementação propriamente dito, este processo divide-se essencialmente na resolução de dois problemas. O primeiro consiste numa efectiva análise do conteúdo armazenado no cubo e o segundo consiste na interpretação e selecção das vistas que efectivamente vão ser guardadas nas estruturas finais [Morfonios et al. 2007].

O primeiro passo, chamado de computação de um cubo, consiste em analisar o esquema de um *data mart* e construir e agrupar os seus agregados, ou seja, construir a *lattice* do cubo de acordo também com as hierarquias estipuladas em cada uma das dimensões associadas ao esquema.

Na figura 2.15 poderá observar-se um exemplo de cubo que possuí três dimensões associadas a uma tabela de factos, a dimensão tempo (t), produto (p) e cliente (c) e a respectiva *lattice* efectuada no passo de computação do cubo.

O ideal seria partir para uma implementação completa da *lattice* do cubo para o servidor OLAP, de modo a poder-se efectuar as *queries* de consulta sobre o cubo com baixos tempos de resposta, mas o espaço de alocação necessário para a sua implementação pode explodir para esquemas dimensionais complexos e com grandes volumes de dados associados [T. Palpanas 2000].

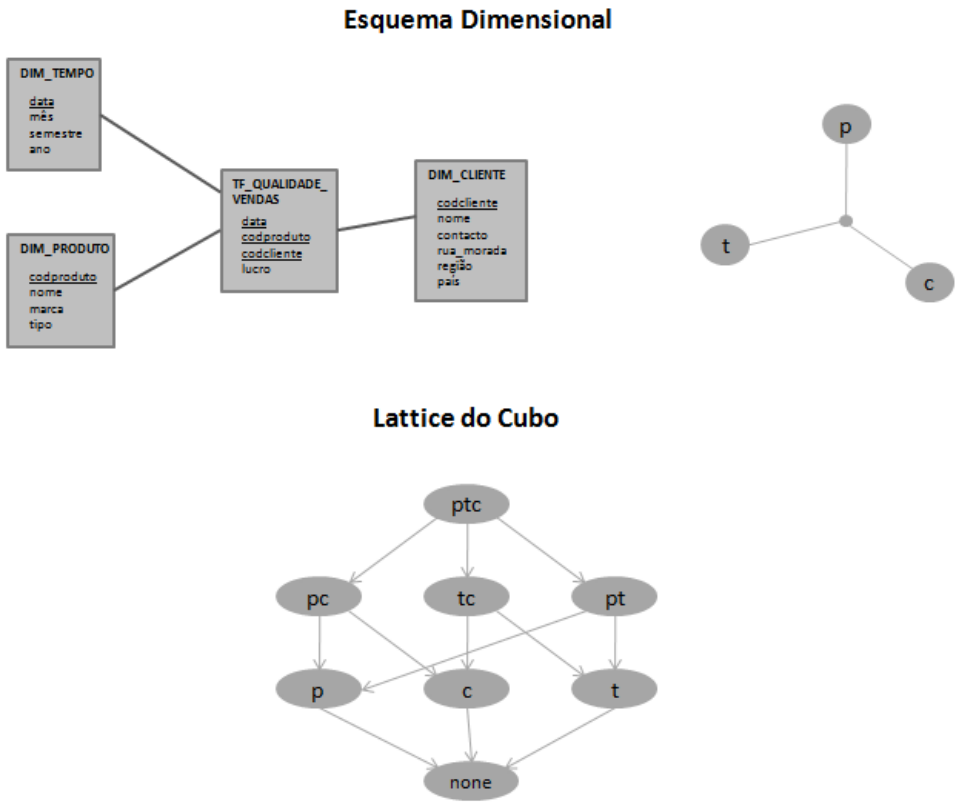

Figura 2.15 - Exemplo de uma *lattice* correspondente a um esquema dimensional.

Nos casos apresentados (figuras 2.15 e 2.16), consegue-se facilmente perceber que o número de *cuboids* gerados na computação de um cubo é muito superior face ao número de dimensões e hierarquias associadas e que portanto, o volume de dados a transpor para o servidor OLAP seria muito maior ao que efectivamente se encontra no *data warehouse*. Para um cubo com *d* dimensões, em que cada dimensão possui um número *L* correspondente aos níveis sobre a hierarquia, então serão gerados o número de *cuboids* correspondente à seguinte fórmula:

$$NCuboides = \prod_{i=1}^{d} (Li + 1)$$

Na figura Fig. 2.16 (Harinarayan et al 1996) é apresentada a *lattice* correspondente para um esquema dimensional em que a tabela de facto possui as dimensões tempo (t), em que se pode

efectuar uma navegação sobre as hierarquias compostas pelo atributo mês (m) e ano (a), em que também uma possui a dimensão cliente (c) com a hierarquia região (r) associada.

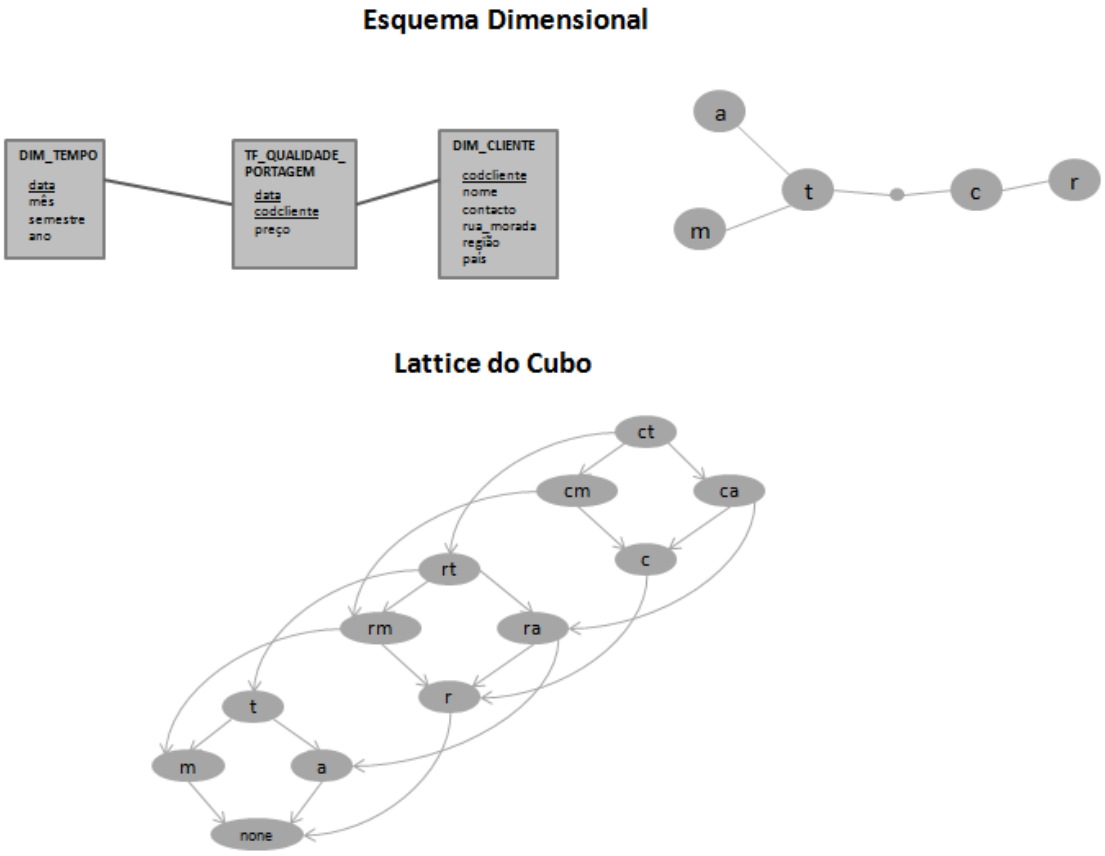

Figura 2.16 - *Lattice* correspondente a um esquema dimensional.

Essencialmente os algoritmos de selecção resolvem o problema de escolher as vistas elaboradas, depois da computação do cubo, que efectivamente devem ser guardadas na fase de implementação do cubo [Lawrence et al. 2006]. Deste modo, os processos de selecção possuem uma importância elevada. Este problema é objecto de intensivos estudos e esforços que têm como objectivo principal o de encontrar os critérios e soluções mais apropriados, para que assim, se determinarem as vistas mais indicadas a materializar [G. Colliat 1996].

De grosso modo, cada algoritmo concentra-se, através do seu modo de funcionamento, em optimizar apenas um dos critérios, mas existem implementações de algoritmos que tentam optimizar mais do que um critério. Nas seguintes subsecções irão abordar-se alguns algoritmos de selecção existentes que tratam este problema, com o objectivo de apresentar alguns raciocínios e operações efectuadas para que deste modo se possa perceber o funcionamento geral de cada um dos algoritmos apresentados.

2.3.1 Visão Geral Sobre Algoritmos de Selecção

Com o objectivo de perceber o funcionamento dos algoritmos de selecção OLAP, encontra-se efectuado em baixo, alguns resumos que pretendem indicar as soluções e mecanismos desenvolvidos pela comunidade científica, que de forma concreta traduzem o grande objectivo que se pretende alcançar: seleccionar as vistas mais indicadas na fase de implementação de um cubo para um servidor OLAP. Mas este propósito tem que ser convertido em critérios objectivos e possíveis de optimizar computacionalmente.

Algoritmo Greedy

O algoritmo Greedy [Harinarayan et al. 1996] foi o primeiro algoritmo onde se tentou proceder uma solução para seleccionar um subconjunto de vistas geradas durante a implementação de um cubo [Morfonios et al. 2007]. Tal como o nome indica este algoritmo baseia-se em métodos *greedy*, em que neste caso particular o objectivo é o de minimizar o tempo necessário para efectuar uma avaliação sobre a *lattice*. Este algoritmo caracteriza-se pelos seguintes factores:

- O número de vistas a materializar é previamente estipulado;
- Os *cuboids* que constituem a *lattice* possuem um custo associado que são dados como parâmetro, como por exemplo o espaço que ocupariam depois de materializados ou a frequência com que são consultados no servidor OLAP, etc;
- O nodo que contém informação sobre todos os atributos (root) é sempre materializado;
- Para um exemplo em que se pretenda materializar *n* vistas, o algoritmo irá efectuar a decisão de identificar quais as *n* vistas a materializar em apenas *n* iterações. Para este caso irão ser materializados os *n* nodos seleccionados, mais o nodo *root*;

Em passos resumidos o algoritmo efectua os seguintes procedimentos [Harinarayan et al. 1996]:

```
S = {root};
FOR (i=1; i<=k; k++) DO
    select that view v not in S such that B(v, S) is maximized;
    S = S U {v};
the greedy selection is S;
```

```
B (v, S) is defined by:
  min = view of least cost in S;
  FOREACH (view u in v) DO
    IF (cost(u) < min) THEN
      sum += cost(u) – min;
  return sum;
```

Algoritmo 1 – Pseudocódigo referente ao algoritmo Greedy.

Algoritmo BPUS

O algoritmo BPUS [Shukla et al. 1998] é uma variação do algoritmo Greedy, seguindo da mesma forma o objectivo principal que se baseia em minimizar o tempo dispendido na selecção sobre as vistas a materializar e tem em conta as mesmas estruturas de *input*. Este algoritmo difere do Greedy principalmente pelo seguinte aspecto, o número de vistas a materializar não é previamente

estipulado, em vez disso tem em conta o espaço máximo disponível que se poderá utilizar ao efectuar a materialização, sem contar com o nodo *root*, dado que esse é sempre materializado.

Em passos resumidos [Shukla et al. 1998] o algoritmo efectua os seguintes procedimentos.

```
S = {root};
A = set that contains all aggregates;
WHILE (space > 0) DO
    select that view v not in S such that B(v, S)/cost(v) is maximized;
    IF (space – space of v > 0) THEN
        space = space – space of v;
        S = S U {v};
        A = A - v;
    ELSE
        space = 0;
S is the set of aggregates picked by BPUS;
```

Algoritmo 2 - Pseudocódigo referente ao algoritmo BPUS.

Desta forma, enquanto houver espaço disponível para seleccionar vistas, o algoritmo vai iterando continuamente até não ser possível efectuar mais junções ao conjunto de vistas seleccionadas. O problema associado com a utilização deste algoritmo é que em muitos casos poderá não efectuar a selecção óptima, dado que poderá cair em situações em que prefira escolher vistas pequenas cuja materialização aparenta trazer grandes benefícios, mas que em termos absolutos seria preferível seleccionar uma outra vista, muito embora, num contexto local, não aparenta ser menos vantajoso [Morfonios et al. 2007].

Algoritmo Greedy-Interchange

O algoritmo Greedy-Interchange [H. Gupta 1997] começa por executar um algoritmo Greedy e sobre o conjunto obtido tenta efectuar uma melhor selecção, iterando sobre o conjunto inicialmente dado. Ele tenta efectuar essa melhoria realizando trocas com vistas que não foram previamente seleccionadas, com a execução do algoritmo inicial. Para isso também recebe como parâmetro os custos associados de cada *cuboid* que constitui a *lattice*. Da mesma forma que o algoritmo BPUS, tem-se em conta o espaço máximo disponível que se poderá utilizar ao efectuar a materialização que é passado como parâmetro [Morfonios et al. 2007].

Em passos resumidos [H. Gupta 1997] o algoritmo efectua os seguintes procedimentos:

```
M = result of running the greedy algorithm;
WHILE (C1 is not null and C2 is not null) DO
    (C1,C2) = pair of views such that C1 is in M and the absolute benefit of (M-C1) U {C2}
is greater than that of M;
    M = (M - C1) union {C2};
M is the set of aggregates picked by Greedy-Interchange;
```

Algoritmo 3 - Pseudocódigo referente ao algoritmo Greedy-Interchange.

Algoritmo Inner-Level Greedy

O algoritmo Inner-Level Greedy [H. Gupta 1997] selecciona também os índices existentes nas vistas escolhidas, que poderão também ser materializadas, caso se conste vantagens com a sua selecção, não fazendo apenas a selecção de vistas a materializar mas também dos índices associados. Da mesma forma que o algoritmo Greedy-Interchange, é executado um algoritmo Greedy e sobre o conjunto obtido tenta efectuar uma melhor selecção iterando sobre o conjunto inicialmente dado, através de trocas sucessivas, tendo em conta o espaço máximo disponível que se poderá utilizar ao efectuar a materialização, que é passado como parâmetro.

Em passos resumidos [H. Gupta 1997] o algoritmo efectua os seguintes procedimentos:

```
M = result of running the greedy algorithm;
space= space constraint;
WHILE (space of M < space)DO
    C = {};
    FOREACH (view vi not in M) DO
        IG = set of vi and some of its indexes selected;
        WHILE (space IG < space) DO
            Lic = Index of vi whose benefit per unit space (M U {IG}) is maximum;
            IG = IG U {Lic};
        IF ( B(IG, M)/S(IG) > B(C, M)/(space of C) ) or ( C = {} ) THEN
            C = IG;
    FOREACH (index Iij such that its view vi in M) DO
        IF ( B(Iij, M)/S(Iij) > B(C, M)/S(C) )THEN
            C = {Iij};
    M = M U {C};

M is the set of aggregates picked by Inner-Level Greedy;
```

Algoritmo 4 - Pseudocódigo referente ao algoritmo Inner-Level Greedy.

De forma resumida o algoritmo infere o benefício obtido por unidade de espaço de *C* a partir da primeira iteração e compara com a dos índices seleccionados nas seguintes etapas. Os melhores são adicionados ao conjunto de vistas a materializar [Morfonios et al. 2007].

Algoritmo r-Greedy

Os algoritmos r-Greedy [Gupta et al. 1997], da mesma forma que o algoritmo Inner-Level Greedy, também selecciona índices existentes nas vistas observadas e executa um algoritmo Greedy para obter um conjunto de partida sobre o qual tenta melhorar, tendo também em conta espaço máximo disponível que se poderá utilizar ao efectuar a materialização (que é passado como parâmetro). A diferença com o algoritmo Inner-Level Greedy é que o número máximo de índices que podem ser seleccionados *r* é limitado, sendo um dos parâmetros de entrada do algoritmo.

Em passos resumidos [Gupta et al. 1997] o algoritmo efectua os seguintes procedimentos:

```
M = result of running the greedy algorithm;
space = space constraint;
r=number of index constraint;
WHILE (space of M < space) DO
    look all sets of one of the following forms:
    - all views not in M and a subset of the indexes (between 0 and r) not in M, or
    - all views in M and all indexes not in M
    C = selected set which has the maximum benefit per unit space in M;
    M = M U {C};
M is the set of aggregates picked by Inner-Level Greedy;
```

Algoritmo 5 - Pseudocódigo referente ao algoritmo r-Greedy.

Segundo testes realizados, parece não valer a pena efectuar o algoritmo devido à complexidade acrescida para $r > 4$, que sobe constantemente quanto maior for o número r estipulado [Morfonios et al. 2007].

Algoritmo MDred-lattice

O algoritmo MDread-lattice [Baralis et al. 1997] difere de todos os algoritmos apresentados até ao momento, dado que selecciona as vistas a materializar na fase de implementação de um cubo tendo em conta um conjunto estipulado de *queries* cujos dados deverão ser guardados. Desta forma optimiza o tempo de respostas das consultas estipuladas como parâmetro, tendo em conta o espaço total que esse conjunto seleccionado de vistas ocuparia no disco.

Em passos resumidos [Baralis et al. 1997] o algoritmo efectua os seguintes procedimentos:

```
SQ = a finite set of queries;
L = nodes of the lattice associated at the SQ;
lastWiews = L;
newViews = null;
WHILE (lastViews is not null) DO
    FOREACH (view vi in lastViews) DO
        FOREACH (view vj in L and vj not in vi) DO
            vx = the most specialized common ancestor between the queries vi and vj;
            IF (vx not in L) THEN
                newViews = newViews U {vx};
    L = L U {newViews};
    lastViews = newViews;
    newViews = null;

L is the set of aggregates picked by MDread-lattice;
```

Algoritmo 6 - Pseudocódigo referente ao algoritmo MDred-lattice.

O algoritmo retorna o conjunto de vistas candidatas a serem materializadas. Inicialmente elabora-se a associação entre os conteúdos que as *queries* abrangem e as vistas correspondentes à *lattice* do cubo gerado. Iterativamente vão sendo adicionados ao conjunto de retorno as vistas que apresentam dependências funcionais entre as vistas previamente seleccionadas [Morfonios et al. 2007].

Algoritmo PBS

O funcionamento do algoritmo PBS [Shukla et al. 1998] é baseado no BPUS, mas o critério para seleccionar a vista a adicionar ao conjunto de retorno é diferente, dado que baseia-se na escolha do *cuboid* que apresenta o menor custo dentro do conjunto de *cuboids* que ainda não foram seleccionados. Da mesma forma que o BPUS o algoritmo recebe como parâmetro o limite máximo de espaço disponível que se poderá utilizar ao efectuar a materialização das vistas seleccionadas.

Em passos resumidos [Shukla et al. 1998] o algoritmo efectua os seguintes procedimentos:

```
S = {root};
A = set that contains all aggregates;
WHILE (space > 0) DO
    select that view v in A but not in S with smallest cost;
    IF (space – space of v > 0) THEN
        space = space – space of v;
        S = S U {v};
        A = A - v;
    ELSE
        space = 0;
S is the set of aggregates picked by PBS;
```

Algoritmo 7 - Pseudocódigo referente ao algoritmo PBS.

Com esta troca de critério, é de notar que a selecção de vistas é feita de baixo para cima da *lattice* (*bottom-up*), atravessando a *lattice* a partir dos *cuboids* com menor custo para os *cuboids* que teoricamente possuem maior nível de detalhe sobre a informação agregada. Embora se obtenha um algoritmo de selecção com melhores tempos de execução, a qualidade dos resultados costumam ser mais baixos [Morfonios et al. 2007].

Algoritmo Inverted-Tree Greedy

O algoritmo Inverted-Tree Greedy [Gupta et al. 1999] destaca-se de todos os algoritmos apresentados até este momento, dado que selecção das vistas a materializar não se baseia apenas pelos factores associados ao espaço máximo, que poderá utilizar na materialização do cubo. O

algoritmo tem em conta o tempo necessário para a manutenção e actualização das vistas seleccionadas. Dado que, enquanto o espaço de armazenamento aumenta á medida que são adicionadas vistas ao conjunto seleccionado, o custo de manutenção associado à selecção de mais uma vista poderá não aumentar. É possível que o custo de manutenção de um conjunto de vistas diminua após a inserção de mais vista ao conjunto, desde que se observem dependências funcionais entre as vistas seleccionadas o que tornaria mais rápido o processo de actualização dos seus descendentes.

Em passos resumidos [Gupta et al. 1999] o algoritmo efectua os seguintes procedimentos:

```
M = {};
Bc = 0;
s = limite time of maintaining cost;
DO
    FOREACH ( inverted tree set of views T in G such that T intercepted with M = {} ) DO
        tm = cost of maintaining views in T with the selected views in M;
        m = cost of maintaining selected views in M ;
        IF ((tm –m) <= s) and (B(T,M)/(tm –m)  > Bc) THEN
            Bc = B(T, M)/tm;
            C = T ;
    M = M U {C};
WHILE (U(M) > s);
M is the set of aggregates picked by Inverted-Tree Greedy;
```

Algoritmo 8 - Pseudocódigo referente ao algoritmo Inverted-Tree Greedy.

O algoritmo iterativamente selecciona um conjunto de vistas ao conjunto de retorno, que apresentam maior benefício em relação aos até então seleccionados, enquanto a restrição associado ao tempo de manutenção associado à manutenção do cubo não for atingida [Morfonios et al. 2007].

Algoritmo PGA

Os algoritmos apresentados até ao momento possuem complexidade polinomial no que diz respeito ao número de *cuboids* presentes numa *lattice*, o algoritmo PGA [Nadeau et al. 2002] apresenta uma solução que resolve o problema de seleccionar as vistas que efectivamente devem ser seleccionadas na fase de implementação do cubo, com uma complexidade polinomial, mas ao nível do número de dimensões existentes no esquema dimensional. O algoritmo PGA divide-se em duas fases distintas, a fase de nomeação e a fase de selecção.

No seguinte esquema [Nadeau et al. 2002] está representado o diagrama de actividades durante a execução do algoritmo:

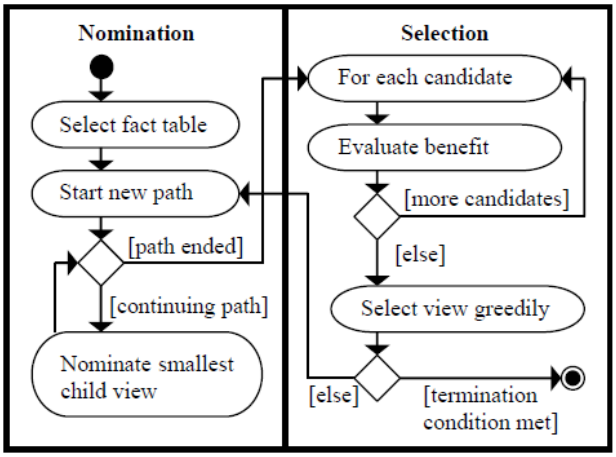

Figura 2.17 – Esquema correspondente à execução do algoritmo PGA.

No esquema acima representado constata-se que o algoritmo evita as duas fontes de complexidade exponencial no que diz respeito ao número de dimensões, dado que durante as iterações não consideram todos os *cuboids* que ainda não foram analisados e que constam na *lattice*, mas também não efectua cálculos associados a todos os *cuboids* descendentes a um previamente seleccionado [Morfonios et al. 2007].

Algoritmo Key

O algoritmo Key [Kotsis et al. 2000] difere de todos os outros apresentados em cima, até mesmo nos critérios utilizados para fazer a selecção das vistas a materializar. Ele executa processos de forma a ignorar os *cuboids* que apenas contem registos redundantes e que desta forma podem ser construídos efectuando uma projecção sobre as vistas que foram então seleccionadas e que desta forma não possuem dados redundantes.

O algoritmo key atravessa a *lattice* gerada de modo *bottom-up* e *breadth-first* procurando as denominadas chaves observacionais (uma chave observacional é similar a uma chave candidata, com a diferença que este tipo de chaves são propriedades referentes aos valores que os dados possuem e não ao nível do esquema de dados). Uma vista, em que o conjunto de atributos é um sobreconjunto de uma chave observacional, então contem o mesmo número de registos que a tabela de factos, dessa forma pode ser reconstruída através de uma projecção a partir da tabela de factos. Um *cuboid* que seja identificado como completamente redundante, implica que todos os seus descendentes também irão possuir a mesma propriedade. Dessa forma consegue-se excluir *cuboids* completamente redundantes e seleccionar apenas os que efectivamente contêm a informação necessária para inferir as restantes vistas que compõe o cubo.

Em passos resumidos [Kotsis et al. 2000] o algoritmo efectua os seguintes procedimentos:

```
i = 0;
s = 0;
K = null;
WHILE ( i < NumberOfCuboids – 1 ) DO
    IF ( GroupBy[i].size < R.size ) THEN
        IF ( GroupBy[i].schema is in K(s) )  THEN
            i = i + 1;
        ELSE IF (found duplicate) THEN
            i = i + 1;
        ELSE
            s = s +1;
            add the GroupBy schema to K(s);
    i = i + 1;

K is the set of aggregates picked by Key;
```

Algoritmo 9 - Pseudocódigo referente ao algoritmo Key.

Testes realizados comprovam que a aplicação deste algoritmo pode poupar até 85% do espaço a utilizar com a materialização das vistas seleccionadas, através do uso do algoritmo Key.

Algoritmo HRU

O algoritmo HRU [Hanusse et al. 2009] é um algoritmo de selecção que tem como objectivo o de conciliar as três medidas de performance que avaliam a execução de um algoritmo de selecção, ou seja a quantidade de memória que utilizam para materializar um determinado cubo, a complexidade do algoritmo e o custo de processar uma *query* sobre um conjunto de *cuboids*. O algoritmo HRU desta forma devolve um conjunto de vistas que em termos absolutos maximiza o benefício associado à sua escolha.

Em passos resumidos [Hanusse et al. 2009] o algoritmo efectua os seguintes procedimentos:

```
space = the amount of memory available;
C = {lattice};
S = {root};
space = space -  space of S;
WHILE (space > 0) DO
    c = cuboid not in S such that B(c, S) is maximized;
    IF (space – space of c > 0) THEN
        space = space –  space of c;
        S = S U {c};
        C = C\{c};
    ELSE
        space=0;
S is the set of aggregates picked by HRU;
```

Algoritmo 10 - Pseudocódigo referente ao algoritmo HRU.

A principal desvantagem deste algoritmo é o tempo despendido para maximizar o benefício, que para um grande número de dimensões tende a ser muito grande. Por cada iteração, precisamos de calcular o benefício de (n^2) *cuboids* com n = 2^D (*D* corresponde ao número de dimensões), o que se torna bastante inviável quando *D* é demasiado grande.

Outros Algoritmos

Para além dos onze algoritmos acima apresentados, existem ainda muitos outros, que possuem também diferentes critérios de selecção, formas de iterar sobre a lattice de um cubo e com diferentes performances de execução. A título de exemplo ficam aqui referências aos algoritmos DynaMat [Kotidis et al. 1999], MVPP (Multiple View Processing Plan), VRDS (View Revelance Driven Selection) [Valluri et al. 2002], IGA, FPUS, o SOMES (uSer Oriented Materialized viEw Selection) [Lin et al. 2007], CBDMVS (Clustering-based dynamic materialized view selection) [Gong et al. 2008], MVA [Lijuan et al. 2009], PSC (Pick Small Cuboids), PTB (Pick the Border), PickBorders [Hanusse et al. 2009] e os algoritmos do tipo Randomized Search [Morfonios et al. 2007] como por exemplo o Random Sampling, o Iterative Improvemente, o Simulated Annealing e o Two-Phase Optimization.

Para além destes algoritmos existem muitos outros, que apresentam outras soluções para a resolução deste problema. A variedade de soluções e mecanismos encontrados reflecte o esforço existente e é prova do intenso trabalho por parte da comunidade científica em encontrar os critérios e os procedimentos mais indicados para que assim se consiga determinar o conjunto de vistas mais apropriados para efectuar a sua materialização na implementação de um cubo, para o servidor OLAP.

2.3.2 Características dos Algoritmos de Selecção

Nas seguintes tabelas encontra-se uma síntese sobre dezanove algoritmos de selecção estudados mais a fundo. Sobre cada algoritmo de selecção constam-se os autores do algoritmo, o ano em que foi publicado, as suas características principais e também a sua complexidade. Desta forma encontram-se resumidas as propriedades e critérios mais relevantes que os algoritmos de selecção utilizam para indicar as vistas a materializar na fase de implementação de um cubo OLAP.

Legenda:
n -> número de *cuboids* presentes numa *lattice*
k1 -> número de vistas seleccionadas pelo algoritmo de selecção
k2 -> número de vistas e índices seleccionadas pelo algoritmo de selecção
T -> T é o número de tuplos nas tabelas de facto presentes no *data warehouse*
d -> número de dimensões de presentes dum esquema dimensional
m -> espaço total da *lattice* de um cubo

Algoritmo	Autores	Ano	Características	Complexidade
Greedy	Venky Harinarayan Anand Rajaraman Jeffrey Ulman	1996	- Algoritmo do tipo Greedy; - Efectua a selecção em k iterações, passadas como parâmetro; - Os *cuboids* que compõe a *lattice* tem previamente um custo associado, é utilizado como critério de escolha as que apresentam menor custo; - Nodo *root* é sempre materializado.	$O(k \cdot n^2)$
Inner-Level Greedy	Himanshu Gupta	1997	- Algoritmo do tipo Greedy; - É executado em primeiro lugar um algoritmo Greedy em que se tenta melhor o resultado obtido; - A selecção baseia-se no espaço máximo (valor passado como parâmetro) e dos índices que as vistas seleccionadas poderão ocupar; - Para além de adicionar vistas ao conjunto de selecção, também pode seleccionar índices associadas às vistas.	$O(k^2 \cdot n^2)$
Greedy-Interchange	Himanshu Gupta	1997	- Algoritmo do tipo Greedy; - É executado em primeiro lugar um algoritmo Greedy em que se tenta melhor o resultado obtido; - A selecção baseia-se no espaço máximo (valor passado como parâmetro) que as vistas seleccionadas poderão ocupar.	A seu tempo de execução pode nunca terminar
r-Greedy	Himanshu Gupta Venky Harinarayan Anand Rajaraman	1997	- É executado em primeiro lugar um algoritmo Greedy em que se tenta melhor o resultado obtido; - A selecção baseia-se no espaço máximo (valor passado como parâmetro) e dos índices que as vistas seleccionadas poderão ocupar; - Para além de adicionar vistas ao conjunto de selecção, também pode seleccionar índices associadas às vistas; - Possui um número máximo de vistas e índices a seleccionar (r), que é passado como parâmetro.	$O(k \cdot n^r)$
MDred-lattice	Elena Baralis Stefano Paraboschi Ernest Teniente	1997	- Recebe como parâmetro um conjunto de *queries* (dadas como prioritárias), em que são analisadas as dependências com a *lattice* do cubo.	$O(n \cdot \log(n))$

Tabela 2.1 - Descrição dos Algoritmos de Selecção OLAP analisados - parte 1.

Algoritmo	Autores	Ano	Características	Complexidade
BPUS	Amit Shukla Prasad Deshpande Jeffrey F. Naughton	1998	- A selecção baseia-se no espaço máximo (valor passado como parâmetro) que as vistas seleccionadas poderão ocupar; - Nodo *root* é sempre materializado; - Seleccionam-se as vistas que apresentam maior densidade de dados.	$O(k \cdot n^2)$
PBS	Amit Shukla Prasad Deshpande Jeffrey F. Naughton	1998	- A selecção baseia-se no espaço máximo (valor passado como parâmetro) que as vistas seleccionadas poderão ocupar; - Nodo *root* é sempre materializado; - Seleccionam-se as vistas que apresentam menor densidade de dados.	$O(k \cdot n^2)$
Inverted-Tree Greedy	Himanshu Gupta Inderpal S. Mumick	1999	- O algoritmo tem em conta o tempo necessário para a manutenção e actualização das vistas seleccionadas; - Utiliza como restrição o tempo associado à manutenção das vistas seleccionadas; - Recebe como parâmetro um limite máximo de tempo de custo de manutenção.	$O(n^2)$
DynaMat	Yannis Kotidis Nick Roussopoulos	1999	- Não indica vistas por inteiro para materializar, mas sim alguns subconjuntos associados às vistas (V); - Efectua uma monitorização sobre as *quieries* de consulta sobre o cubo de dados e infere as vistas mais utilizadas; - Divide o seu processamento em duas fases, a fase *in-line* e a fase *update*.	Complexidade na fase de *update* do cubo é $O(V^2)$
Key	Nikolaos Kotsis Douglas McGregor	2000	- Não selecciona as vistas em que os valores podem ser deduzidos através das vistas previamente seleccionadas; - O conjunto de vistas seleccionadas não possui dados redundantes; - Analisa a *lattice* do cubo e efectua a opção de selecção caso não possua uma chave observacional dentro das vistas até então seleccionadas.	$O(T \cdot n)$
PGA	Thomas P. Nadeau Toby J. Teorey	2002	- Possui complexidade polinomial ao nível do número de dimensões existentes no esquema dimensional; - Por cada iteração efectua dois processos, a nominação e a selecção.	$O(d \cdot k^2 \cdot n)$

Tabela 2.2 - Descrição dos Algoritmos de Selecção OLAP analisados - parte 2.

Algoritmo	Autores	Ano	Características	Complexidade
VRDS	S. R. Valluri Soujanya Vadapalli K. Karlapalem	2002	- A selecção baseia-se no espaço máximo (valor passado como parâmetro) e dos índices que as vistas seleccionadas poderão ocupar; - Mantém uma matriz onde constam as vistas mais relevantes, deduzidas através das *queries* que usualmente consultam os dados do cubo; - Selecciona as vistas com maior pontuação dentro da matriz enquanto a restrição do espaço limite não for atingida.	
SOMES	Ziyu Lin Dongqing Yang Guojie Song Tengjiao Wang	2007	- Tem em conta as características das *queries* de consulta e os perfis de utilizadores que consultam um cubo; - Recebe como parâmetro a composição dos perfis de utilizadores, as respectivas vistas que usualmente consultam e as *queries* de consulta que efectuam; - Actualiza as vistas a seleccionar face aos perfis de utilização dos grupos e das *queries* submetidas.	$O(k \cdot n^2)$
CBDMVS	An Gong Weijing Zhao	2008	- Analisa a semelhança entre as *queries* de consulta efectuadas sobre o cubo; - Selecciona as vistas correspondentes face aos conteúdos identificados como sendo os mais relevantes.	$O(n^2 .\log(2) .n)$
MVA	Zhou Lijuan Ge Xuebin Wang Linshuang Shi Qian	2009	O seu processamento divide-se em três fases principais. Primeiro selecciona-se as vistas face às *queries* de consulta que são mais vezes efectuadas. Na segunda parte actualizam-se as vistas seleccionadas tendo em conta a probabilidade das *queries* que se identificam como mais prováveis. A terceira parte consiste na projecção de processos que reduzem os custos das actualizações no servidor.	
PSC	Nicolas Hanusse Sofian Maabout Radu Tofan	2009	- Recebe como parâmetro o limite máximo para que um *cuboid* possa ser seleccionado; - Selecciona as vistas que não ultrapassam o limite estipulado, enquanto não for ultrapassado o máximo de espaço possível de materializar no servidor OLAP.	$O(2^d)$

Tabela 2.3 - Descrição dos Algoritmos de Selecção OLAP analisados - parte 3.

Algoritmo	Autores	Ano	Características	Complexidade
HRU	Nicolas Hanusse Sofian Maabout Radu Tofan	2009	- A selecção baseia-se no espaço máximo (valor passado como parâmetro) que as vistas seleccionadas poderão ocupar; - Nodo *root* é sempre materializado; - Tenta seleccionar as vistas em que em termos absolutos maximiza o benefício associado; - Para esquemas com muitas dimensões a execução do algoritmo torna-se prolongada.	Por cada iteração analisa-se $O((2^d)^2)$ *cuboids*
PTB	Nicolas Hanusse Sofian Maabout Radu Tofan	2009	- Recebe como parâmetro o limite máximo de espaço para que um *cuboid* possa ser seleccionado; - Selecciona as vistas que não ultrapassam o limite estipulado, efectuando a travessia dividida pelas diversas dimensões, podendo assim não atravessar a *lattice* inteira.	$O(d \cdot 2^d)$
PickBorders	Nicolas Hanusse Sofian Maabout Radu Tofan	2009	- Executa o algoritmo PTB, passando como parâmetro os valores possíveis que estipulam o limite máximo de espaço a ser utilizado. - Vai adicionando ao conjunto seleccionado as diferenças encontradas.	$O(\log(m).d.2^d)$

Tabela 2.4 - Descrição dos Algoritmos de Selecção OLAP analisados - parte 4.

2.3.3 Critérios de Optimização

Tal como explicado anteriormente, os algoritmos de selecção OLAP desempenham a função de seleccionar as vistas mais indicadas na fase de implementação de um cubo com vista a encontrar o devido equilíbrio entre os recursos disponíveis e o tempo de resposta das *queries* de consulta efectuadas sobre o cubo. Este equilíbrio é deveras importante, dado que se não forem transpostas para o servidor OLAP as vistas que possuem uma maior utilização, o servidor terá que constantemente consultar o *data warehouse*, converter os dados nas estruturas indicadas para servir as ferramentas de *front-end* e só depois enviar os resultados requeridos.

O objectivo principal, embora seja fácil de definir e seja intuitivo entender o contexto em que os algoritmos de selecção OLAP são aplicados, torna-se um problema quando se tenta proceder à sua resolução recorrendo a processos computacionais. O objectivo principal tem que traduzido ou decompostos em critérios de simples implementação.

Sintetizados os critérios estudados com a análise dos algoritmos acima abordados, lista-se de seguida os métodos de selecção identificados:

- Iteração sobre uma *lattice* associada a um cubo, sendo previamente atribuídos custos, que traduzem um critério que se pretende minimizar ou maximizar.
- Analisar uma *lattice* de um cubo tendo em conta o limite de espaço que as vistas seleccionadas poderão ocupar no, servidor OLAP.
- Analisar uma *lattice* de um cubo tendo em conta o esforço necessário para proceder à actualização dos valores correspondentes ao cubo de dados no servidor OLAP.
- Seleccionar as vistas a materializar tendo em conta a frequência das *queries* de consulta sobre o cubo, durante a sua utilização.
- Seleccionar índices associados às vistas observadas, de forma a obter melhores tempos de execução através do aproveitamento deste tipo de associações.
- Selecção de partes de vistas, em vez de proceder à indicação total das vistas para materializar.
- Recurso a estruturas auxiliares, de forma a proceder à identificação das vistas mais utilizadas e consultadas sobre o cubo.
- Recurso a formulas (probabilísticas em certos casos) e parâmetros numéricos de entrada, que indicam a traduzem a decisão de optar por materializar uma determinada vista para o servidor OLAP.

Capítulo 3

Selecção de Hiper-Cubos com Base em Padrões de Exploração OLAP

Durante os últimos anos, as ferramentas analíticas vieram dar um contributo enorme na evolução das plataformas de análise, na sua sofisticação e versatilidade, bem como na facilidade de acesso e de cruzamento de dados. Os sistemas de processamento analítico (OLAP – ON-Line Analytical Processing) apresentam hoje reflexos sérios dessa evolução, o que fez com que se tornassem um dos principais actores em cenários típicos de suporte à decisão. Podemos ver um sistema OLAP composto, essencialmente, por três componentes: um *data warehouse*, um servidor OLAP e um conjunto de ferramentas de acesso e análise de dados [Cuzzocrea et al. 2009]. Neste último componente, temos as ferramentas de *front-end*, que se ligam a servidores OLAP contendo uma selecção prévia dos dados presentes no *data warehouse*, selecção esta usualmente mantida em estruturas multidimensionais de dados denominadas por hiper-cubos, ou simplesmente por cubos.

A "passagem" total dos dados presentes num *data warehouse* para um servidor OLAP torna-se inviável (ou mesmo impossível) na grande maioria dos casos [T. Palpanas 2000], dado que os cubos correspondentes contêm todas as agregações possíveis, realizadas de acordo com as hierarquias estipuladas dentro de cada dimensão do esquema dimensional [Sapia et al. 1998]. É, então, na fase de implementação do cubo, a partir de um *data warehouse* para o servidor OLAP, que são seleccionadas as vistas que efectivamente deverão ser materializadas [Lawrence et al. 2006]. As vistas que não sejam seleccionadas para materialização poderão na mesma serem consultadas. Todavia, estas terão tempos de acesso substancialmente maiores, uma vez que o servidor OLAP terá que recolher no *data warehouse* correspondente os dados pretendidos pelas consultas sobre elas realizadas, converte-los para a estrutura correcta e finalmente enviar os resultados para a ferramenta de acesso que requisitou tal informação. Assim, vemos facilmente que o processo de selecção das vistas de um cubo possui grande importância no estabelecimento do devido equilíbrio entre os recursos disponíveis no servidor OLAP e os tempos de resposta

referentes às consultas efectuadas sobre o cubo. Uma das formas possíveis para realizar uma selecção precisa sobre os dados que efectivamente são consultados, é através da monitorização permanente das *queries* que são efectuadas pelos utilizadores OLAP e tentar perceber quais as vistas que eles mais solicitam [Lin et al. 2007] durante cada uma das suas sessões OLAP. Desta forma, na fase de implementação do cubo, existe uma noção bastante concreta de quais as vistas mais relevantes para o sistema OLAP, podendo-se assim atribuir algum tipo de prioridade a cada uma delas durante o processo de selecção.

Neste capítulo, propõe-se um esquema de selecção de vistas, o método M3, que tem como estrutura base uma cadeia de Markov [R. Sarukkai 2000] [Ching et al. 2006] especialmente concebida para reflectir o nível de utilização de cada utilizador sobre as vistas de um determinado cubo. Cada nodo da cadeia de Markov, gerada a partir da informação relacionada com todas as sessões OLAP monitorizadas, corresponde a uma vista (ou um conjunto de vistas), tendo a si associado uma cor reflectindo a frequência de consulta. A partir daqui, e aplicando um conjunto específico de filtros - restrições de espaço e de processamento -, desencadeia-se o processo de selecção do método.

Neste capítulo abordar-se-á os algoritmos de selecção estudados que inspiraram o processo de concepção e desenvolvimento do método M3. Adicionalmente, com o intuito perceber o funcionamento geral do método M3 irá explicar-se os principais processos executados na selecção das vistas mais frequentemente consultadas. Após o método de selecção ser abordado de forma sucinta, proceder-se-á a uma apresentação mais técnica e detalhada sobro o método, analisando de forma específica a sua execução passo a passo. De forma a entender o método M3 num caso real, apresenta-se também um pequeno caso de estudo em que, com uma demonstração prática e ilustrativa, se pode analisar os processos executados.

3.1 Planeamento do Método de Selecção

Uma das maiores preocupações que os investigadores de processamento analítico têm desde há muito tempo está relacionada com o problema da selecção de vistas de hiper-cubos. Este problema tem sido alvo de inúmeros estudos em muitos cenários aplicacionais e reside, basicamente, em saber qual o melhor conjunto de vistas multidimensionais de dados a materializar, de forma a maximizar a satisfação das *queries* dos utilizadores e minimizar a tempo de processamento e as necessidades de armazenamento de um qualquer hiper-cubo numa plataforma analítica.

Na literatura do domínio encontramos vários métodos e estratégias para fazer a selecção de vistas. Uma das formas mais efectivas de o fazer é através da utilização da informação associada com as *queries* multidimensionais dos utilizadores ao longo do tempo durante vários dos seus períodos de exploração dos hiper-cubos – as sessões OLAP. A exploração dessa informação pode-nos conduzir ao estabelecimento de perfis de exploração OLAP e, consequentemente, ao estabelecimento de fronteiras de materialização – condições *icebergue*.

No método proposto nesta dissertação – o método M3 - apresentamos e descrevemos uma forma de selecção de vistas de hiper-cubos, especialmente orientado para a definição de tais condições de materialização, reflectindo as diversas sessões OLAP estudadas a partir de um conjunto de cadeias de Markov. As cadeias de Markov, devido à informação que possuem e à forma como está organizada, permitem facilmente aplicar conceitos e executar travessias sobre a rede, uma vez que existe um enorme leque algoritmos e conceitos associados aos diagramas de Markov já avançados pela comunidade científica [Coffman et al. 1985].

Para a criação do método M3, foram estudados vários algoritmos de selecção, com o intuito de perceber o funcionamento geral dos métodos até agora avançados, mas principalmente para identificar características que possibilitem acrescentar operacionalidade e eficiência no momento de planeamento dos processos nele incorporados, assim como para perceber como se efectua a validação e cálculo do seu desempenho durante a sua execução.

Uma forma de proceder a uma selecção precisa sobre os dados que efectivamente são consultados, é efectuar uma constante monitorização das *queries* de consulta e tentar perceber quais as vistas que os agentes de decisão observam mais frequentemente durante as sessões OLAP [Lin et al. 2007]. Desta forma, na fase de implementação de um cubo, conseguimos ter uma noção bastante precisa de quais as vistas mais relevantes e atribuir a sua prioridade durante a etapa da sua selecção.

3.1.1 O Cubo e as Condições de Iceberg

Na fase de implementação de um cubo de dados são executados vários processos com o objectivo de escolher um conjunto de vistas para serem materializadas num servidor OLAP. Uma forma de escolher esse conjunto de vistas é estabelecer condições, denominadas de condições de *iceberg*, de forma a poderem ser definidos processos que recolham as partes da *lattice* do cubo que verifiquem as condições estabelecidas [Beyer et al. 1999]. O denominado "problema *Icegerg-CUBE*" consiste em efectuar a computação de todas as agregações que verificam uma condição de selecção, tal como uma cláusula *HAVING* de uma *querie* SQL.

De forma a entender a execução de uma condição de iceberg sobre um cubo de dados, poderíamos definir um método de selecção que tem uma condição que estipule que cada vista seleccionada deva possuir pelo menos um número *N* de tuplos - este parâmetro *N* é usualmente designado por suporte mínimo, ou *minsup.* Um cubo *iceberg* com um *minsup* é facilmente expresso em SQL com uma cláusula *CUBE BY*.

```
SELECT A, B, C, D, COUNT(*), SUM(M)
FROM FT
CUBE BY A, B, C, D
HAVING COUNT(*) >= N
```

Algoritmo 11 – Exemplo condição de *iceberg* expressa num comando SQL.

Num contexto real esta abordagem é muito simplista, dado que não existe um estudo sobre a forma como os dados estão organizados, como estão constituídos, ou quais os dados que possuem maior incidência de consulta. Mas, de facto, todos os métodos de selecção executam processos deste género para que no final, da mesma forma que o método de selecção em cima definido, retornem um subconjunto de vistas do cubo, o que na prática forma um *iceberg* quando analisamos a *lattice* de um cubo como um todo.

Na figura 3.1 encontra-se ilustrada uma *lattice* de um cubo, em que cada *cuboid* possui um valor associado que indica o número de tuplos que esse mesmo *cuboid* possui.

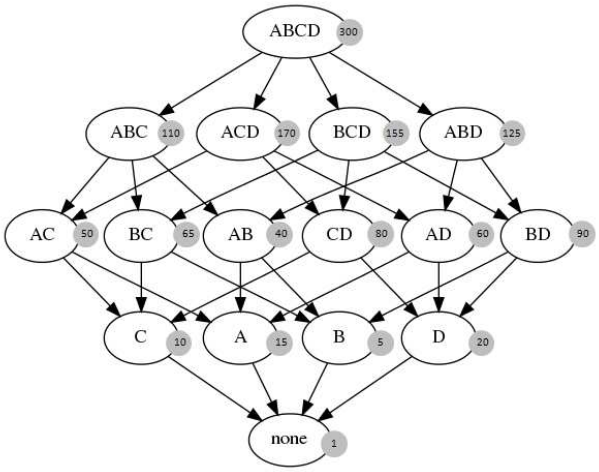

Figura 3.1 - Exemplo de uma *lattice*, com pesos associados a cada *cuboide*.

Se sobre a *lattice* aplicarmos o método de selecção acima definido com um *minsup* igual a 80, o resultado que se obtém para a selecção de cuboides encontra-se representado na figura 3.2, em que se verifica que os *cuboids* que satisfazem a condição encontram-se pintados a cinzento.

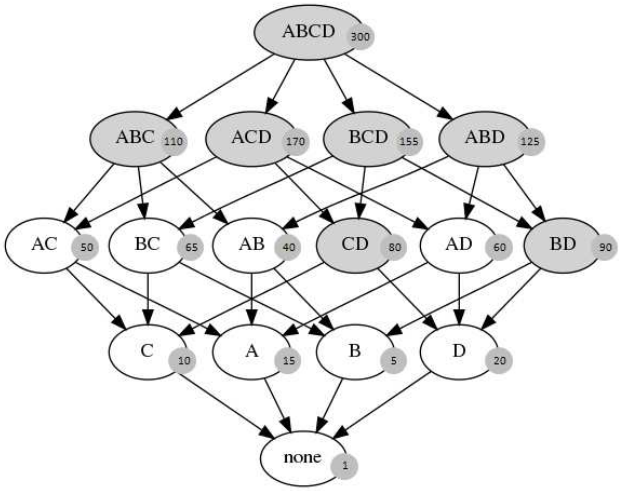

Figura 3.2 - Exemplo de um cubo *iceberg*.

Dado que o armazenamento integral dos dados presente no *data warehouse* num servidor OLAP torna-se impraticávell em diversos casos, a motivação para desenvolver processos que levem à construção de cubos *iceberg* é de facto muito grande. É necessário encontrar formas concretas que possibilitem seleccionar partes do cubo a serem materializadas tendo em conta determinados os critérios que são estipulados por cada método. Desta forma, quanto mais adequadas forem as condições geradas pelos métodos de selecção, ou seja, condições que definam o devido equilíbrio entre os recursos disponíveis e o tempo de resposta das consultas efectuadas, melhores serão os desempenhos das ferramentas analíticas de dados no memento das sessões de consulta.

3.1.2 Características Adoptadas dos Algoritmos Estudados

O método M3 divide-se em três processos principais. O primeiro processo consiste na coloração da *lattice* de acordo com as restrições de análise aplicadas sobre o cubo e a sua utilização ao longo das suas várias sessões de consulta. Este primeiro processo baseou-se em algumas características já definidas em outros algoritmos de selecção avançados pela comunidade científica. Estipulou-se que o método M3 deveria possuir quatro parâmetros de entrada, que condicionariam o conjunto de vistas a devolver pelo método.

Para construir a *lattice* colorida do cubo, de acordo com a frequência de consulta de cada um dos *cuboids* que compõe o cubo, desencadearam-se os seguintes processos e definiram-se as seguintes características, que foram inspirados a partir de outros algoritmos:

- Fez-se a atribuição prévia do montante de espaço que cada *cuboid* do cubo irá ocupar caso seja materializado no servidor OLAP. Esta atribuição é importante, dado que um dos parâmetros de entrada do método de M3 é precisamente o montante máximo que o conjunto de vistas a ser indicado poderá possuir no limite.

- Foi definido a atribuição de um índice a cada um dos *cuboids* que compõe o cubo de dados que reflecte a sua frequência de consulta ao longo do período de tempo indicado. Desta forma, encontra-se criado um atributo chave, dado que o que se pretende obter no final de todo o processo, as vistas mais frequentemente utilizadas ao longo das sessões OLAP efectuadas sobre o cubo.

- Foram definidas estruturas auxiliares, a serem integradas posteriormente no servidor OLAP, de forma a serem armazenadas informações relativas às consultas efectuadas sobre o cubo de dados e assim identificar as vistas mais frequentemente consultadas.

- Recorreu-se a fórmulas para determinar qual o grau de frequência de consulta de um determinado *cuboid* de dados e filtros que possuem um valor estático de forma a efectuar a decisão que leve à indicação que se deve materializar uma determinada vista ou não.

Muito embora a maior parte dos processos que compõem o M3 se tenham planeado sem tirar partido de nenhum outro método de selecção, o primeiro passo do método M3 (a coloração da *lattice* do cubo) foi influenciado a partir de outros algoritmos de selecção existentes. As características aproveitadas constituem uma mais-valia que contribui para a qualidade do conjunto

de vistas devolvido pelo método, dado que de forma rigorosa possibilitam efectuar a associação dos quatro parâmetros de selecção submetidos ao método M3 com as características de cada uma das vistas que compões o cubo de dados.

3.1.3 Características do Método de Selecção Elaborado

Durante a fase de implementação de um cubo de dados realizam-se alguns processos com o objectivo de escolher o conjunto de vistas mais adequado para materializar num servidor OLAP, de acordo com esta ou aquela perspectiva de utilização. A materialização de todas as vistas (*cuboids*) de um cubo é algo que na prática não é muito viável, tanto em termos de processamento das estruturas multidimensionais como do seu posterior armazenamento e consequente exploração. Uma das formas possíveis para escolher esse conjunto de vistas, é através do estabelecimento de algumas condições especiais, usualmente denominadas por condições *iceberg*, que possam regular a selecção dos *cuboids* (ou uma parte) que integram a *lattice* de um dado cubo [Beyer et al. 1999]. Com base nessas condições, estabelecidas através da definição de algum critério de materialização ou descobertas através da análise da exploração do cubo, conseguimos reduzir de forma muito significativa as vistas não utilizadas e que seriam, sem a aplicação desses critérios, materializadas para nada.

Com esse mesmo objectivo em vista, desenvolvemos um método de selecção para cubos *iceberg*, que desenvolve os seus critérios de materialização com base em informação de exploração que recolhe através da monitorização constante da utilização de um dado cubo de dados, por parte da sua comunidade de utilizadores. Este método, que designámos simplesmente por M3 (método de selecção em três fases) utiliza um sistema de coloração que reflecte na *lattice* do cubo segundo a frequência com que cada um dos seus *cuboids* é acedido. Os vários *cuboids* vão assumindo uma coloração mais avermelhada à medida que vão sendo sucessivamente consultados e permanecem em tons de azul conforme não sejam pesquisados. No final de um dado período de análise a *lattice* do cubo terá um espectro de cores tal que revelará a forma como o cubo foi explorado – cores vermelhas significam maior número de consultas.

Após esta atribuição de cores, que reflecte a utilização de cada um dos *cuboids*, são aplicados os quatro parâmetros de entrada do método M3, que são:

- o intervalo de tempo de análise das consultas efectuadas sobre o cubo durante as sessões OLAP;
- as dimensões que os *cuboids* deverão ser constituídos para pertencerem ao conjunto de selecção;
- o limite mínimo de frequência de selecção;
- o limite máximo de espaço que o conjunto de vistas seleccionadas poderá possuir no limite.

Desta forma, cria-se um espectro de selecção composto por estas quatro variáveis. Caso todas elas atinjam o seu valor máximo significa que é dada indicação para materializar o cubo por completo, mas caso se redefinam o valor de qualquer uma das variáveis, significa que se pretende diminuir o conjunto de vistas seleccionado, havendo a possibilidade de fazer cortes/ajustes segundo cada uma destas variáveis de análise.

Na figura 3.3 encontra-se ilustrado o espectro definido pelas quatro variáveis de análise, cujo valor é o máximo permitido em cada um dos quatro casos.

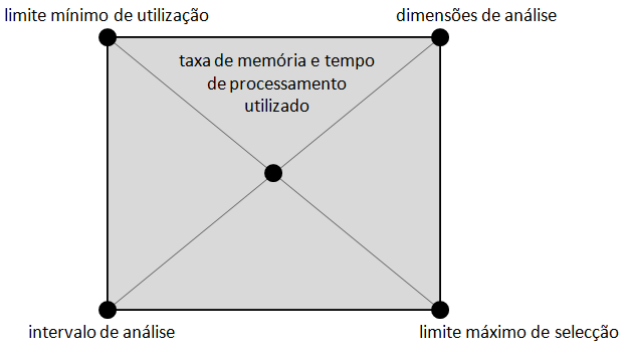

Figura 3.3 - Espectro de análise composto pelos quatro parâmetros de selecção.

Na figura figura 3.4 representa-se a situação em que é dada a indicação de que o limite mínimo de frequência que as vistas seleccionadas deverá possuir, deve ser reduzido. Esta indicação resulta numa poupança imediata de memória a ser dispendido no servidor OLAP, assim como o tempo de processamento a ser despendido pelo do método, uma vez que irão ser descriminadas as vistas que não verifiquem o limite mínimo de utilização redefinido.

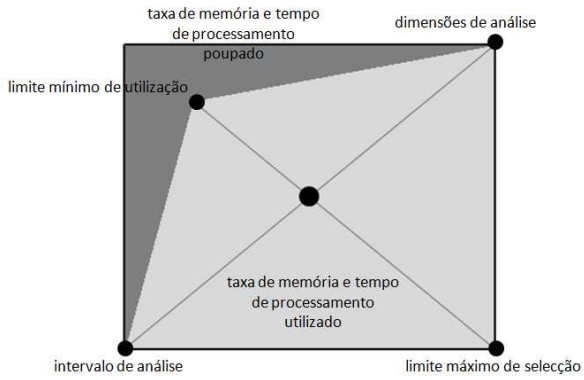

Figura 3.4 - Espectro de análise com o parâmetro "limite mínimo de utilização" redefinido.

Á medida que se redefinam os quatro parâmetros (figura 3.5) mais memória e tempo de processamento se conseguirá poupar. A contrapartida que existe, quando se redefine cada um dos quatro parâmetros, é que no caso de se efectuar atribuições demasiado restritivas haverão vistas que, embora sejam frequentemente utilizadas, poderão não fazer parte do conjunto seleccionado.

Assim, a atribuição destes valores deve ser efectuada de forma ponderada e que possibilite a efectiva selecção das vistas mais indicadas, ou seja as vistas que mais frequentemente são consultadas durante as sessões OLAP.

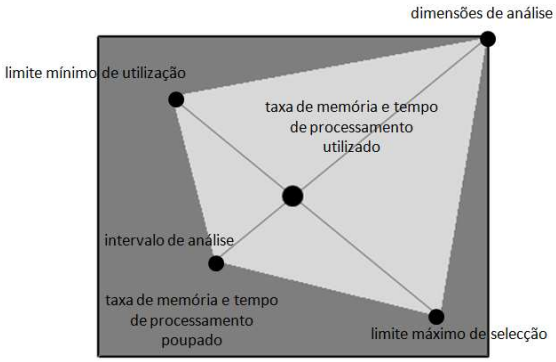

Figura 3.5 - Espectro de análise composto pelos quatro parâmetros de selecção redefinidos.

3.2 M3, Computação de Cubos Icebergue a Partir de Sessões OLAP

O objectivo principal do método M3 é o de seleccionar, de forma precisa, os *cuboids* que mais frequentemente são consultados, desenvolvendo assim uma garantia mais forte que os dados que se encontram no servidor OLAP, estão de acordo com as pretensões dos agentes de decisão, o que resulta na pretendida regulação entre os recursos disponibilizados no servidor e as performances das sessões OLAP. A única forma de efectuar a correspondência das vistas mais consultadas sobre um cubo de dados, é efectuar uma monitorização continua das vistas consultadas sobre o cubo. Desta forma, na fase de implementação de um cubo, existe uma precisa noção de quais as vistas mais relevantes, podendo atribuir prioridades durante o passo de selecção. O método M3 propõe um esquema de selecção de vistas que tem como estrutura base uma cadeia de Markov correspondente às vistas pesquisadas durante as sessões de consulta, em que cada nodo da cadeia corresponde a uma vista (ou um conjunto de vistas) que tem a si associada uma cor referente à frequência com que foi consultada. Sobre o diagrama são aplicados filtros e restrições de forma a seleccionar as vistas mais consultadas e desta forma as que possuem maior probabilidade de serem consultadas posteriormente.

O M3 divide-se essencialmente em três passos principais, em que cada passo é executado com o intuito de preparar as estruturas de dados para que na última fase do método se consiga obter a indicação do conjunto de vistas de maior importância e que será posteriormente consultado pelos agentes de decisão, evitando assim ao máximo que haja a necessidade de consultar o *data warehouse* para satisfazer as consultas a efectuar. Na primeira fase do método M3 analisa-se os parâmetros de entrada e de forma correspondente constrói-se a *lattice* do cubo colorida de acordo com a frequência de consulta sobre cada vista de dados. Em segundo plano, elabora-se uma

cadeia de Markov que organiza as sequências de consulta efectuadas ao longo das sessões de exploração e reúne-se essa a informação com a execução do primeiro processo. Desta forma é elaborada uma cadeia de Markov colorida, em que fica composta a estrutura final sobre a qual se aplicam processos que irão efectuar a selecção das vistas a materializar, na última etapa do processo.

3.2.1 Monitorização das Sessões OLAP

O processo de monitorização das sessões de consulta efectuado sobre o cubo consiste essencialmente em acrescentar um canal no servidor OLAP no qual a *query* de consulta é interpretada e são armazenadas informações sobre a consulta efectuada. Na figura figura 3.6 encontra-se ilustrado o processo de monitorização das *queries* submetidas a um cubo de dados durante as sessões OLAP.

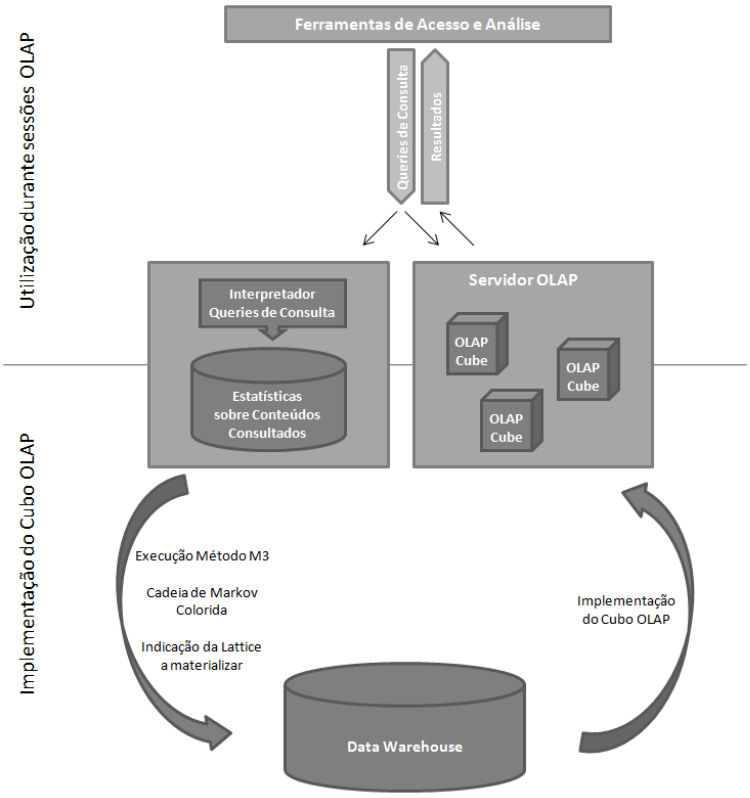

Figura 3.6 - Processo de monitorização das sessões OLAP e implementação do Cubo.

A *query* é, portanto, replicada, seguindo o seu curso normal de consulta e resposta. Todavia, por outro lado existe um processo adicional no qual se efectua um estudo da *query* e o registo de informações referentes à consulta. Por cada *query* submetida é registada a vista pesquisada (ou o

conjunto de vistas no caso de retornar mais que uma vista), o instante em que se efectuou a consulta e algumas indicações sobre a sequência em que a consulta se insere na sessão OLAP.

No momento de implementação do cubo no Servidor OLAP é dada indicação que é necessário obter o conjunto de vistas mais frequentemente consultadas. Nesse momento, é executado o método M3, que começa por corresponder os parâmetros de entrada com as informações recolhidas ao longo das sessões de consulta e executa os três processos que irão levar à selecção das vistas que os agentes de decisão mais procuram, durante as sessões de consulta OLAP.

3.2.2 Processos Executados pelo Método M3

O método M3 assenta as suas decisões de modo a calcular o conjunto final de vistas seleccionado, tendo como base a cadeia de Markov colorida, a estrutura sobre a qual se reúne a informação retirada a partir da monitorização constante das consultas efectuadas sobre o cubo dados e que sobre a qual são executados os processos que seleccionam as vistas a materializar.

O M3 efectua três grupos de processos que se denominam, nomeadamente, por:

- **Coloração da *lattice* do cubo** - Esta fase do processo tem como principal objectivo o de atribuir a cada vista que constitui a *lattice* do cubo a cor associada com a frequência de utilização verificada ao longo das várias sessões OLAP, de acordo com quatro condições previamente estabelecidas, nomeadamente: o intervalo de tempo de análise, as dimensões a analisar, o limite mínimo de utilização e o limite máximo de selecção. No final desta etapa, todos os *cuboids* sem qualquer cor são aqueles que não foram alvos de qualquer *query* multidimensional. Por outro lado, os *cuboids* com as cores mais avermelhadas foram aqueles que mais *queries* receberam nas sessões OLAP monitorizadas.

- **Coloração da Cadeia de Markov** - Nesta segunda fase, faz-se a geração da Cadeia de Markov reflectindo as várias sessões OLAP efectuadas pelos utilizadores analíticos sobre o cubo alvo, que nos permite ver a forma como cada *cuboid* foi consultado e as várias sequências de consulta seguidas pelos utilizadores. De seguida, ainda nesta fase, fazemos reflectir a informação relativa à *lattice* colorida na fase anterior na Cadeia de Markov aqui gerada e obtemos uma nova estrutura que denominámos por Cadeia de Markov colorida. A estrutura agora obtida, combina as características da *lattice* colorida com as características base de um Cadeia de Markov, o que nos permite obter as sequências de *queries* mais frequentes (mais coloridas) e *cuboids* coloridos envolvidos. Mais uma vez, a intensidade da coloração revela-nos a importância do caminho escolhido na cadeia e dentro dele a importância do *cuboid* relativamente ao processo.

- **Construção do Cubo Iceberg** - Por fim, na terceira fase, definem-se as vistas a materializar. A forma como o fazemos assenta, essencialmente, na Cadeia de Markov colorida, aplicando-se três restrições base que determinam quais as vistas que "deverão" ser materializadas tendo em consideração o nível de utilização OLAP. As três restrições referidas

têm como função identificar as vistas que possuem pouca relevância no contexto das consultas OLAP efectuadas sobre o cubo [Ching et al. 2006].

Coloração da Lattice do Cubo

Para que a aplicação do método M3 tenha alguma efectividade, todas as sessões OLAP devem ser monitorizadas. Assim, na altura da computação do cubo, os dados recolhidos durante o processo de monitorização são analisados e a *lattice* do cubo é colorida, de acordo com o número de vezes que cada uma das vistas (neste caso os *cuboids*) foi consultada, e a respectiva Cadeia de Markov gerada, com base nas sequências de *queries* seguidas durante as várias sessões OLAP. O esquema de coloração da *lattice* (e posteriormente das Cadeias de Markov coloridas) segue o esquema de cores apresentado na figura 3.7, em que as cores azuladas são utilizadas para tonificar as vistas menos utilizadas e as cores avermelhadas para tonificar as vistas mais consultadas.

Figura 3.7 - Esquema de cores usado para colorir a *lattice* do cubo de dados.

Assim, o primeiro processo executado sobre a *lattice* do cubo, corresponde em colorir as vistas que o compõe, de acordo com as seguintes condições estabelecidas inicialmente:

- **Intervalo de tempo em que as sessões de consulta foram efectuadas** - Esta restrição é útil, dado que se houver a noção exacta dos instantes em que as vistas foram consultadas, poder-se-á efectuar estudos tendo como base períodos de tempo;
- **Dimensões a serem analisadas** - Pode existir a necessidade de excluir dimensões na análise sobre a *lattice*, que inicialmente não se pretendem materializar no servidor OLAP. Desta forma constata-se ser útil definir uma restrição que possa excluir dimensões sobre o conjunto de análise;
- **Limite mínimo de utilização** - De forma a serem dispensadas da análise as vistas que não foram consultadas o suficiente, de acordo com um limite de frequência mínimo, torna-se útil poder definir essa mesma restrição. Assim garante-se que as vistas que no final são seleccionadas foram consultadas o número de vezes suficiente para estarem no conjunto de selecção;
- **Limite máximo de selecção** - Em vários algoritmos de selecção, é estabelecido o parâmetro que traduz o limite que o conjunto de selecção não pode ultrapassar [Morfonios et al. 2007]. Este limite pode-se referir por exemplo ao espaço máximo que o servidor OLAP disponibiliza para a materialização do cubo, pode-se referir ao esforço necessário para proceder à actualização dos valores correspondentes ao cubo de dados no servidor OLAP, ou entre outros critérios possíveis, que se pretendam limitar.

Com a execução deste primeiro passo, associam-se cores a cada uma das vistas que verifiquem as condições acima indicadas, mas também as vistas que se encontram exactamente entre duas

vistas candidatas a serem seleccionadas, mas que não foram seleccionadas/coloridas. As vistas que não possuem nenhuma cor associada, nunca irão pertencer ao conjunto de selecção final.

Coloração da Cadeia de Markov

Na segunda fase do método M3 é onde que se realiza a conjugação da informação subjacente à *lattice* colorida do cubo de dados e a Cadeia de Markov gerada para as sessões OLAP consideradas. Desta forma, conseguimos fazer a coloração da Cadeia de Markov, que depois de colorida permitirá identificar a frequência com que foi consultada cada uma das vistas que figuram nos diversos nodos da Cadeia de Markov gerada. Cada nodo que integra a Cadeia de Markov colorida, refere-se a uma vista da *lattice* do cubo de dados, mas que aqui foi pintada de acordo com a sua frequência de utilização. Por seu lado, cada ramo possui a percentagem do número de vezes que foi efectuada a transição entre as vistas que o suportam.

Esta estrutura possui propriedades muito interessantes no que se refere à representação das sessões de consulta efectuadas sobre o cubo [R. Sarukkai 2000]. Com ela tem-se a possibilidade de efectuar um estudo preciso relativamente à forma como o cubo é utilizado pelos seus utilizadores e determinar com bastante efectividade, para um mesmo conjunto prático de cenários aplicacionais, as vistas mais indicadas a materializar na altura da implementação do cubo. A construção da Cadeia de Markov colorida constitui o último passo antes de executar os processos de determinação das vistas indicadas para materialização.

Selecção do Conjunto de Vistas

Efectuada a construção do diagrama de Markov colorido, entra-se no último passo do método de selecção. Este passo consiste na selecção das vistas presentes no diagrama de Markov colorido aplicando três processos que atravessam a rede e efectuam a eliminação de ligações e nodos tendo em conta três restrições. As três restrições estipuladas, têm como função identificar vistas que possuem pouca relevância no contexto das consultas OLAP efectuadas sobre o cubo [Ching et al. 2006]. As três restrições são as seguintes:

- Eliminação das ligações que envolvam relações com vistas não seleccionadas/coloridas na primeira parte do método de selecção. Com a aplicação desta restrição, garante-se que os nodos não seleccionados ficam isolados e portanto não serão objecto de análise nas futuras travessias sobre a rede.
- Eliminação das ligações em que não se verifiquem uma probabilidade de execução maior do que 30%. A aplicação desta restrição garante a exclusão das ligações que possuem uma baixa probabilidade de execução, podendo acontecer, eventualmente, o isolamento de vistas que irão ser eliminadas posteriormente.
- Eliminação dos nodos da rede que não se encontrem em nenhum caminho em que se verifique a possibilidade de atravessar a partir do nodo de início (representado com a letra "I") e fim (representado com a letra "E") de uma determinada sessão de consulta.

A aplicação destes três processos na geração de um diagrama de Markov colorido, apenas com as vistas que possuíram uma maior incidência durante as sessões de OLAP, uma vez que os processos desencadearam a descriminação das vistas, que não possuíam consultas suficientes para se serem dadas como vantajosas de serem materializadas, mas também porque, no contexto das sessões de consulta, não se inserem em sequências significativamente relevantes.

Com a aplicação dos dois primeiros processos, pode ocorrer uma situação em que se quebrem todas as ligações entre o inicio e o fim das sessões de consulta sobre o cubo, declaradas no diagrama de Markov colorido. Nesse caso, ao aplicar a terceira restrição, o diagrama ficará com todas as vistas eliminadas, dado que durante as sequências das sessões OLAP levadas a cabo pelos agentes de decisão, não se exprimiu uma incidência forte sobre um conjunto determinado de vistas. Dessa forma, o conjunto de vistas retornadas pelo método de selecção será composto pelas vistas coloridas no primeiro passo do método, em que já se aplicaram métodos que realçam as vistas tendo em conta as quatro características estipuladas e relacionadas com intervalo de tempo em análise, dimensões de análise, limite mínimo de utilização e o limite máximo de selecção.

3.2.3 Definição do Método M3

Após efectuado todo o planeamento a nível da informação a captar durante as sessões de consulta OLAP. Depois de ficarem definidos os parâmetros de entrada que regulam o nível de esforço a efectuar pelo método M3, de forma a seleccionar o conjunto de vistas a materializar. Uma vez estabelecidos os principais processos que o método a implementar deveria de possuir, de forma a alcançar a qualidade de resultados que se ambiciona. Passou-se finalmente à implementação do método, com o objectivo de efectuar casos de estudo, mas também para efectuar comparações a nível do desempenho com outros algoritmos de selecção.

De seguida apresentamos em pseudocódigo os processos que o método M3 possui implementados. Esta definição em pseudocódigo foi organizada de acordo com o principal grupo de processos que o método M3 executa e tem como objectivo apresentar um breve resumo dos principais passos que são executados nos processos do M3.

Tal como referido anteriormente, começa-se por definir os valores dos quatro parâmetros de entrada, nomeadamente o intervalo de tempo de análise (*ia*), o conjunto de dimensões a analisar (*da*), o limite mínimo de utilização (*lmu*) e o limite máximo de selecção (*lms*). A partir daí, são efectuados os três grupos de processos nos quais consecutivamente se redefinem as estruturas de dados de acordo com a atribuição de diferentes prioridades de selecção e são aplicados os filtros sobre essas mesmas estruturas, de forma a efectuar decisões que seleccionam o conjunto de vistas óptimo a indicar.

No bloco de pseudocódigo correspondente ao algoritmo 1 podemos encontrar a definição do método central do algoritmo de selecção implementado. Como se pode observar primeiro é elaborada a *lattice* do cubo colorida, de seguida é crida a cadeia de Markov colorida. Após a elaboração da cadeia são aplicadas as três restrições que efectuam a selecção do conjunto de vistas a materializar.

```
M3 (ia, da, lmu, lms) definido por:
        l = latticeCuboColorida(ia, da, lmu, lms);
        s = sessoesConsulta(ia);
        cmc = cadeiaMarkovColorida(l, s);

        r1 = eliminarVistasNaoColoridas(cmc);
        r2 = eliminarLigacoesPoucaProbabilidade(r1);
        r3 = eliminarNodosIsolados(r2);

        ci = cuboIceberg(r3);

        IF(isEmpty(ci))
                return l;
        ELSE
                return ci;
```

Algoritmo 12 - Pseudocódigo referente aos processos principais do método M3.

```
latticeCuboColorida(ia, da, lmu, lms) definido por:
        l = {};
        sum = 0;
        HASH cuboid_utilization = cuboidsPerTimesUtilized(ia);
        HASH cuboid_color = {};
        maxu = getMaxUtilization(cuboid_utilized);

        FOREACH(cuboid c IN lattice)
                IF(c IN da)
                        color = calculateColor(cuboid_utilization.get(c), maxu);
                        IF(color >= lmu)
                                cuboid_color.put(c, color);
                        ELSE IF(isBetweenTwoImportantCuboids(c))
                                cuboid_color.put(c, BETWEEN);
                        ELSE
                                cuboid_color.put(c, TRANSPARENT);
                ELSE
                        cuboid_color.put(c, TRANSPARENT);

        FOREACH(cuboid c IN sortByColor(cuboid_color))
                color = cuboid_color.get(c);
                IF((sum < lms) && (color != TRANSPARENT))
                        sum += cost(c);
                        putInLattice(l, c, color);
                ELSE
                        putInLattice(l, c, TRANSPARENT);

        return l;
```

Algoritmo 13 - Pseudocódigo referente à construção da *lattice* do cubo colorida.

Na área de texto algoritmo 2, encontramos o pseudocódigo correspondente ao método onde se elabora a *lattice* do cubo colorida. Nesta fase começa-se por encontrar a vista do cubo que foi mais utilizada. De seguida é correspondida para cada uma das vistas a sua cor relativamente a sua utilização com a vista mais utilizada. No caso de um vista não for de encontro com os quatro parâmetros estipulados no início da execução do algoritmo de selecção, então não se fará a atribuição de uma cor sobre a vista e portanto nunca fará parte do conjunto de selecção.

```
sessoesConsulta(ia) definido por:
        s = {};
        paths = getPathQueries(ia);

        FOREACH(path p IN paths)
                putPathInChain(s, p);

        return s;
```

Algoritmo 14 - Pseudocódigo referente à construção do diagrama de Markov.

```
cadeiaMarkovColorida(l, s) definido por:
        cmc = {};

        FOREACH(node n IN s)
                color = getColor(n, l);
                putNodeInChain(cmc, n, color);

        return cmc;
```

Algoritmo 15 - Pseudocódigo referente à construção do diagrama de Markov colorido.

Nos blocos de pseudocódigo correspondentes ao algoritmo 3 e 4 pode-mos observar como são constituídos os métodos que elaboram a cadeia de Markov colorida. Cada uma das consultas efectuadas ao longo das sessões OLAP serão marcadas na cadeia de Markov com a percentagem de vezes em que a sequência de consulta se verificou. Cada nodo da cadeia de Markov possuirá a cor correspondente à sua frequência de utilização.

```
eliminarVistasNaoColoridas(cmc) definido por:
        r1 = {};

        FOREACH(path p IN cmc)
                IF((p->node_origin != TRANSPARENT) && (p->node_destination != TRANSPARENT))
                        putPathInChain(p, r1);

        return r1;
```

Algoritmo 16 - Pseudocódigo referente à aplicação da primeira restrição.

```
eliminarLigacoesPoucaProbabilidade(r1) definido por:
        r2 = {};

        FOREACH(path p IN r1)
                IF(p->probability >= 0.30)
                        putPathInChain(p, r2);

        return r2;
```

Algoritmo 17 - Pseudocódigo referente à aplicação da segunda restrição.

```
eliminarNodosIsolados(r2) definido por:
        r3 = {};

        FOREACH(node n IN r2)
                IF(!(isIsolated(n, r2)))
                        putInChain(n, r3);

        return r3;
```

Algoritmo 18 - Pseudocódigo referente à aplicação da terceira restrição.

Nos blocos de pseudocódigo definidos nos algoritmos 5, 6 e 7, encontramos os métodos que executam as três restrições que efectuam a selecção das vistas. Estes métodos correspondem à última fase do M3. Na área de texto correspondente ao algoritmo 8, é elaborada a *lattice* do cubo que deverá ser materializada no processo de implementação do cubo de dados no servidor OLAP.

```
cuboIceberg(r3) definido por:
        ci = {};

        FOREACH(cuboid c IN lattice)
                IF(exists c IN r3)
                        putInLattice(ci, c, color);

        return ci;
```

Algoritmo 19 - Pseudocódigo referente à construção da *lattice* com as vistas a materializar.

3.3 O Caso de Estudo

De forma a entender melhor a aplicação do método M3, seleccionou-se um pequeno caso de estudo no qual se podem verificar cada uma das fases executadas pelo método de selecção implementado e, assim, perceber como é obtido o resultado final com a sua execução, como também tentar compreender a sua dinâmica com recurso a ilustrações de cada um dos processos.

Neste caso, optou-se por um efectuar um esquema dimensional de pequena dimensão, bem como realizar poucas sessões OLAP sobre o cubo alvo. Desta forma, é possível verificar de forma concreta o desenvolvimento de cada um dos processos na prática. Nesse sentido, decidiu-se ilustrar cada um dos três grupos de processos do método M3, ou seja, revelando gradualmente a *lattice* colorida, a cadeia de Markov colorida e, por fim, a aplicação das três restrições sobre a cadeia de Markov que definem o conjunto de selecção de vistas indicadas pelo método M3.

3.3.1 Definição do Caso de Estudo

De forma a justificar o resultado final obtido com o método M3, definiu-se um esquema dimensional com uma tabela de factos (*ft*) e três tabelas de dimensão (figura 3.8):

- Dimensão tempo (*t*) que pode ser seguida em três hierarquias pelos meses (*m*) registados, seguido do ano (*a*), ou da estação (*e*), ou do semestre (*s*).
- Dimensão cliente (*c*), que pode ser agregada pelas regiões (*r*).
- Dimensão produto (*p*), que pode ser agregada pela descrição (*d*).

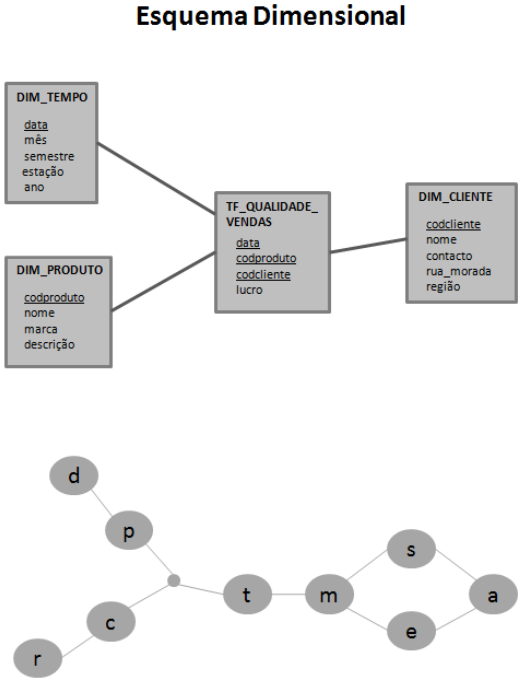

Figura 3.8 - O esquema dimensional utilizado como base de trabalho.

Os esquemas dimensionais de dados, como o apresentado na figura 3.8, são por natureza multidimensionais, uma vez que integram, como o próprio nome indica, diversas dimensões que caracterizam e suportam as várias perspectivas de análise associadas. Além disso, ao definirmos algum tipo de hierarquia, acrescentamos também outras formas de analisar as dimensões. Tudo isto, serve para definir a forma como o cubo correspondente deverá ser gerado (assumindo-se aqui que o esquema apresentado dará origem a um único cubo) e consequentemente as suas diversas vistas materializadas.

Se gerarmos a *lattice* do cubo associado ao esquema dimensional apresentado, obtemos a estrutura multidimensional que apresentamos na figura 3.9. Como podemos ver, a *lattice* gerada integra cinquenta e quatro *cuboids* associados. Isto permite-nos ver o que acontece em termos de volume de dados com a passagem de um esquema dimensional plano (ainda no *data warehouse*) para uma estrutura multidimensional real. Mais uma vez, releva-se aqui a importância dos processos de selecção de vistas como a forma mais equilibrada de conter esta "explosão" de dados e reduzir os recursos computacionais envolvidos, em casos nos quais um grande número de vistas não é simplesmente utilizado.

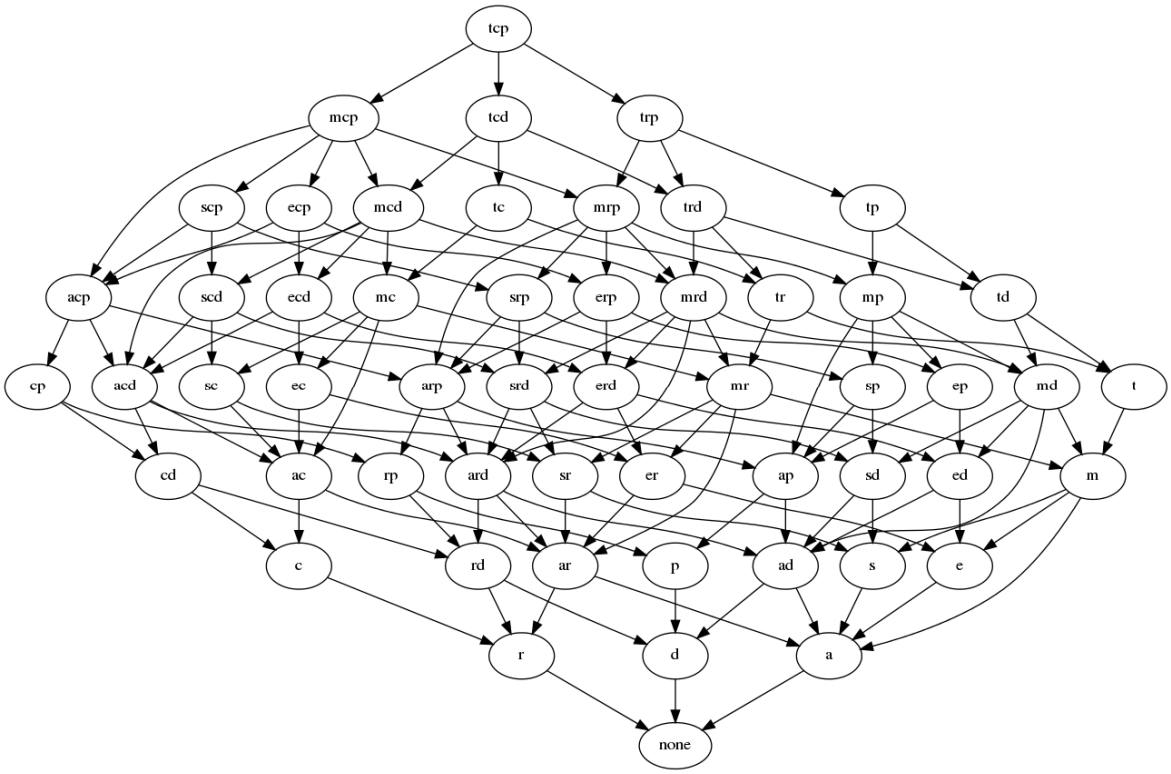

Figura 3.9 - *Lattice* do cubo correspondente ao esquema dimensional da figura 3.8.

3.3.2 Construção da Lattice do Cubo Colorida

Neste passo, em particular, associam-se as cores a cada uma das vistas que verifiquem as condições acima indicadas, mas também às vistas que se encontram exactamente entre duas vistas candidatas a serem seleccionadas, mas que não foram seleccionadas/coloridas. As vistas que não possuem nenhuma cor associada, nunca irão pertencer ao conjunto de selecção final.

Iniciámos o processo definindo uma sessão OLAP sobre o cubo de dados. Nessa sessão OLAP, como podemos verificar pela figura 3.10, foram efectuadas oito operações de consulta.

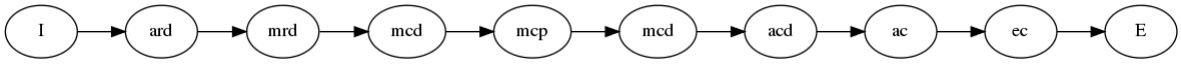

Figura 3.10 - Ilustração da primeira sessão OLAP efectuada sobre o cubo.

Desta forma, se dermos indicação para construir a *lattice* do cubo colorida, sem aplicar qualquer tipo de restrição através dos quatro parâmetros de entrada do método M3 (figura 3.11), verificamos que o *cuboid* "mcd" foi o mais consultado (duas vezes, como se pode verificar na figura 3.10). Por outro lado, os *cuboids* a azul são aqueles que não foram consultados.

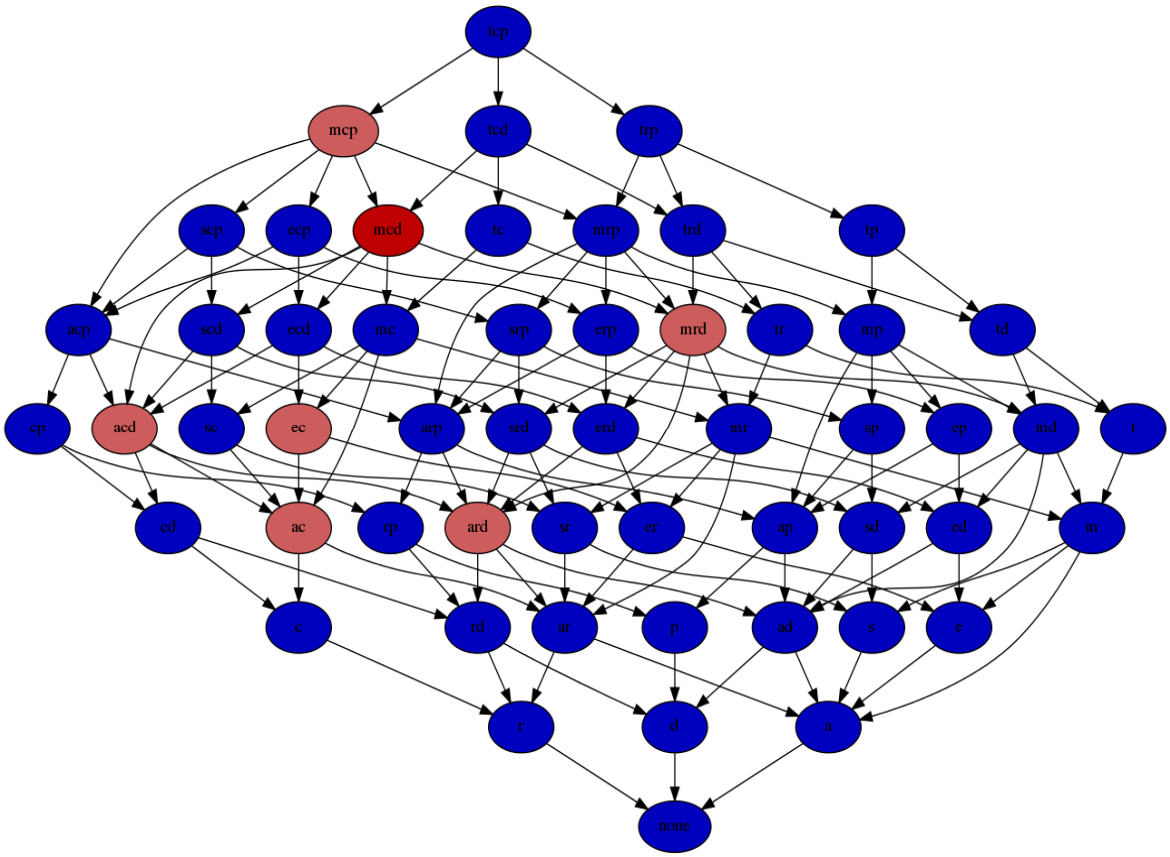

Figura 3.11 - *Lattice* do cubo colorida após realizada a primeira sessão de consulta.

Após efectuadas mais duas sessões de consulta OLAP (figura 3.12) sobre o cubo de dados. Da mesma forma que no caso anterior, não houve qualquer tipo de redefinição dos quatro parâmetros de entrada sobre o método M3. Pode-se observar o resultado da *lattice* do cubo colorido ilustrado na figura figura 3.13. Como se pode verificar, o *cuboid* "mcd" continua a possuir o tom mais avermelhado, dado que foi o que se consultou mais vezes ao longo das três sessões de consulta (quatro ao todo).

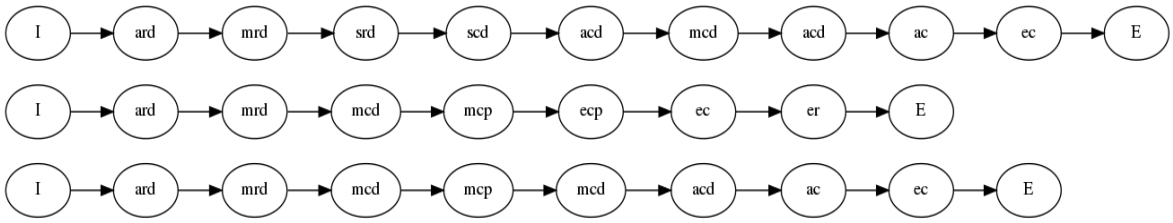

Figura 3.12 - Ilustração da outras sessões OLAP realizadas sobre o cubo.

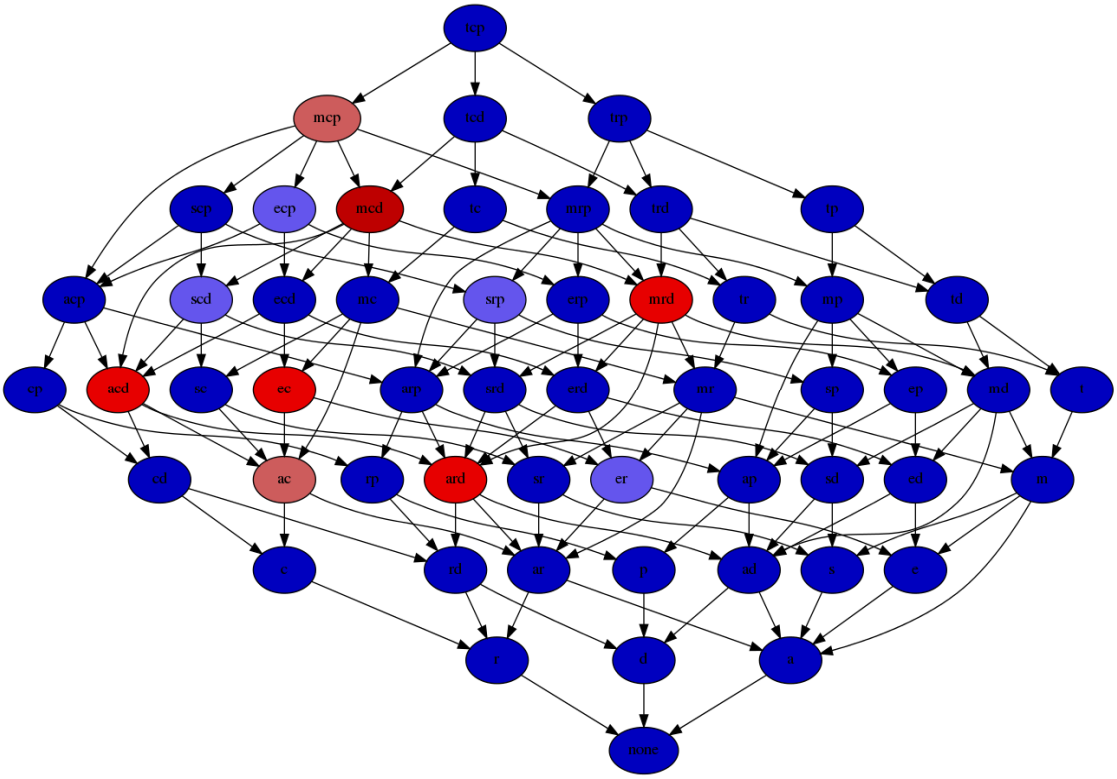

Figura 3.13 - *Lattice* do cubo colorida após efectuadas as três primeiras sessões de consulta.

Como pudemos ver, ao todo, foram efectuadas cinco sessões de consulta sobre o cubo de dados, que apresentamos em grupo na figura 3.14.

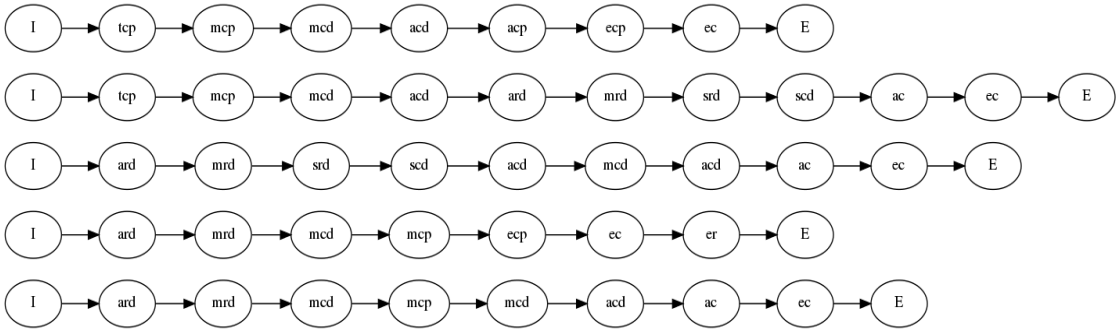

Figura 3.14 - Representação das cinco sessões OLAP efectuadas sobre um cubo de dados.

Na figura 3.15 está representado a *lattice* do cubo após aplicado o primeiro processo do método de selecção. Sobre o cubo foram efectuadas cinco sessões de consulta, e repectivas restrições. Estas restrições apresentam as seguintes características:

- Intervalo de tempo: desde a implementação do esquema dimensional até ao instante em que se executou a computação do cubo de dados.
- Dimensões: não foram excluídas dimensões sobre a análise realizada.
- Limite mínimo de utilização: foram excluídas as dimensões com menos de 50% de utilização, relativamente à vista na qual se verificou o maior número de consultas;
- Limite máximo de selecção: esta restrição relaciona-se com o peso total que o conjunto seleccionado deverá ocupar no servidor OLAP. A *lattice* no total tem o peso de 3114 GB. Mas estipulou-se o limite de 1090 GB para se garantir uma redução de pelo menos 65%.

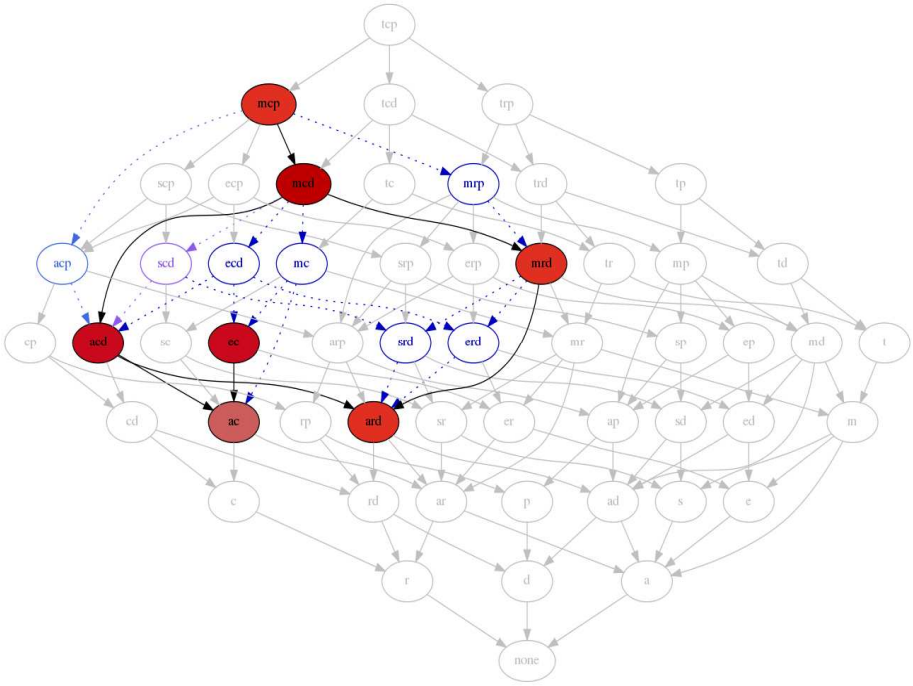

Figura 3.15 - Exemplo do resultado obtido com a execução do primeiro processo.

Os nodos que se encontram coloridos na figura 3.15 são os nodos candidatos a fazerem parte do conjunto de selecção e, como tal, verificam a condição de *iceberg* estipulada. Os nodos que possuem ligações tracejadas e que se encontram preenchidos a branco, também são nodos candidatos a fazerem parte do conjunto de selecção, mas que, apesar disso, não verifiquem a condição de *iceberg* - estes encontram-se entre dois nodos que verificam tais condições e pode tornar-se vantajoso analisá-los mais aprofundadamente nos seguintes processos do método de selecção. Os nodos acinzentados, são nodos que não fazem parte do conjunto de vistas candidatas e portanto nunca farão parte do conjunto de selecção final.

3.3.3 Construção da Cadeia de Markov Colorida

As cinco sessões ilustradas na imagem figura 3.14 permitiram fazer a geração da cadeia de Markov representada na figura figura 3.16. As vistas que compõe esta cadeia de Markov foram coloridas de acordo com a sua frequência de consulta, resultando numa cadeia de Markov colorida. Sobre ela, irão ser executados três processos com o intuito de eliminar as vistas que, no contexto das sequências das sessões OLAP, não possuam uma grande probabilidade de serem consultadas futuramente.

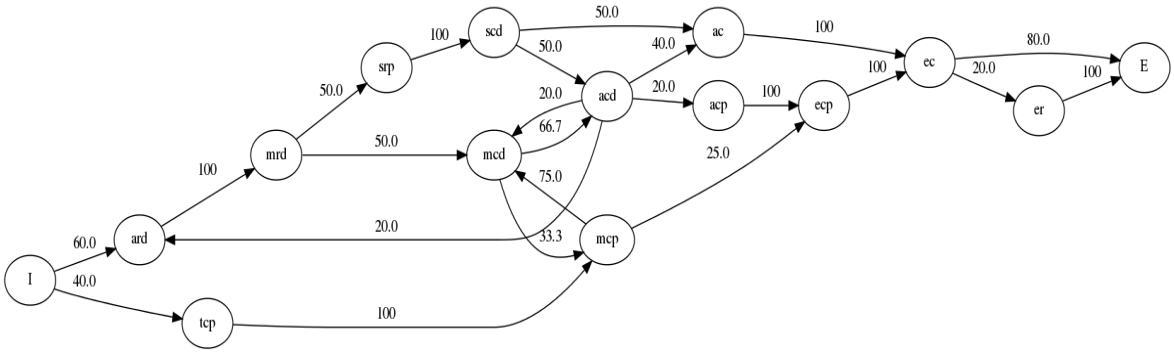

Figura 3.16 - Diagrama de Markov que ilustra as sequências seguidas nas sessões de consulta.

Cada nodo que constitui a cadeia de Markov colorida refere-se a uma vista presente na *lattice* do cubo de dados, mas agora colorida de acordo com a sua frequência de utilização ao longo das várias sessões OLAP realizadas. Cada ligação possui um valor que é a percentagem do número de vezes que foi efectuada a transição entre as vistas envolvidas. A figura 3.17 mostra o resultado obtido com o cruzamento da *lattice* colorida representada na figura 3.15 e a cadeia de Markov representada na figura 3.16.

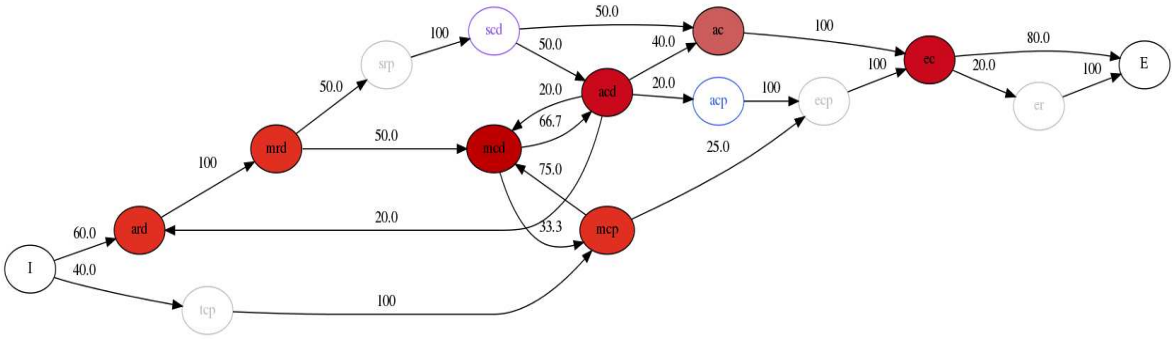

Figura 3.17 - Cadeia de Markov colorida após execução do segundo passo do método M3.

No último passo do M3, sobre esta estrutura de dados vão ser aplicados três filtros, que irão, por fim, determinar qual o conjunto de vistas a seleccionar para serem materializadas no servidor OLAP.

3.3.4 Selecção do Conjunto de Vistas

A selecção do melhor conjunto de vistas com base na Cadeia de Markov colorida, constitui a última fase de aplicação do método proposto nesta dissertação. Nesse sentido, são aplicados sobre a cadeia gerada três processos, que em termos gerais, fazem a eliminação de ligações e de nodos, tendo em conta três restrições bastante concretas, com o objectivo de identificar todas as vistas que possuem pouca relevância no contexto das consultas efectuadas sobre o cubo [Beyer et al. 1999]. Essas restrições são as seguintes:

1) eliminação das ligações que envolvam relações com vistas não seleccionadas/coloridas na primeira parte do método de selecção;

2) eliminação das ligações em que não se verifiquem uma probabilidade de execução maior do que 30% - de referir que, este valor foi arbitrariamente definido, podendo assumir qualquer outro valor entre 0 e 100%, que corresponde à taxa de materialização pretendida;

3) eliminação dos nodos da rede que não se encontrem em nenhum caminho no qual se verifique a possibilidade de atravessar a partir do nodo de início ("I") e o nodo de fim ("E") de uma determinada sessão de consulta.

O resultado da aplicação destes três processos é um Cadeia de Markov colorida, apenas com as vistas que possuam a maior incidência de consulta durante as sessões de OLAP. Nas figuras 3.18,

3.19 e 3.20 encontram-se exemplificadas as aplicações sucessivas de cada uma das três restrições enunciadas sobre o esquema dimensional definido anteriormente.

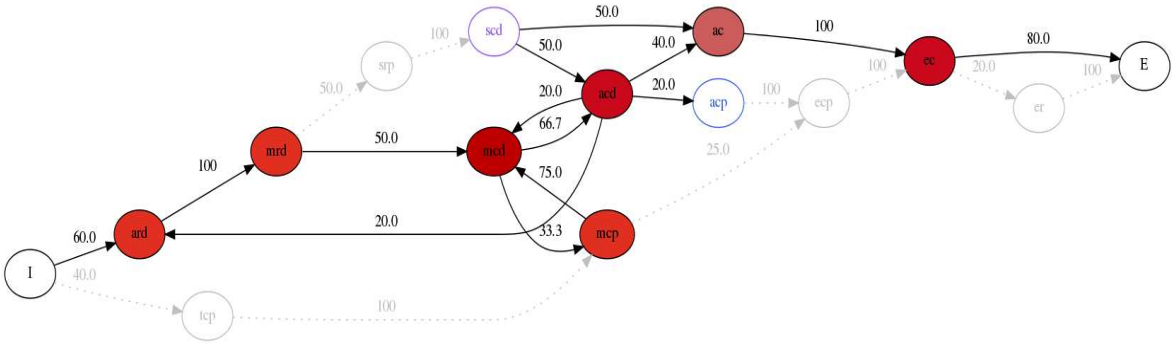

Figura 3.18 - Reflexo da aplicação da primeira restrição.

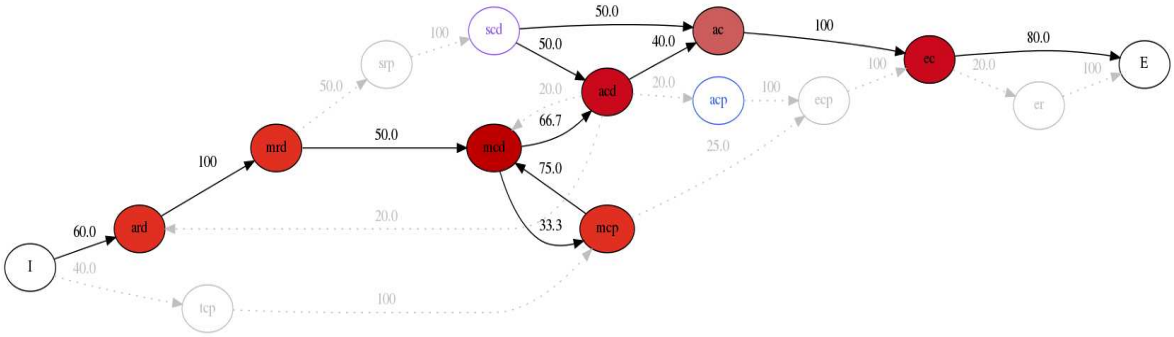

Figura 3.19 - Resultado da aplicação da segunda restrição.

Observando a cadeia representada na figura 3.18, constata-se que dois dos nodos que nela figuram não se encontram coloridos e, como tal, devem ser isolados do resto da rede. Agora, se repararmos na figura 3.19, verificamos que os ramos que possuem uma probabilidade inferior a 30% foram eliminados, tal como é requerido pela segunda restrição enunciada. Por fim, com a aplicação da terceira restrição, discriminámos um nodo do diagrama, uma vez que este não está presente em nenhum caminho entre o estado inicial e o estado final da Cadeia de Markov colorida. No final, o método de selecção desprezou seis vistas, dentro das treze vistas que realmente foram utilizadas durante as sessões OLAP. De realçar o facto de que este método incide preferencialmente sobre as vistas mais frequentemente consultadas durante as sessões OLAP, o que sugere um conjunto de vistas a materializar que vão de encontro com as preferências dos utilizadores e com a informação que eles mais frequentemente analisam.

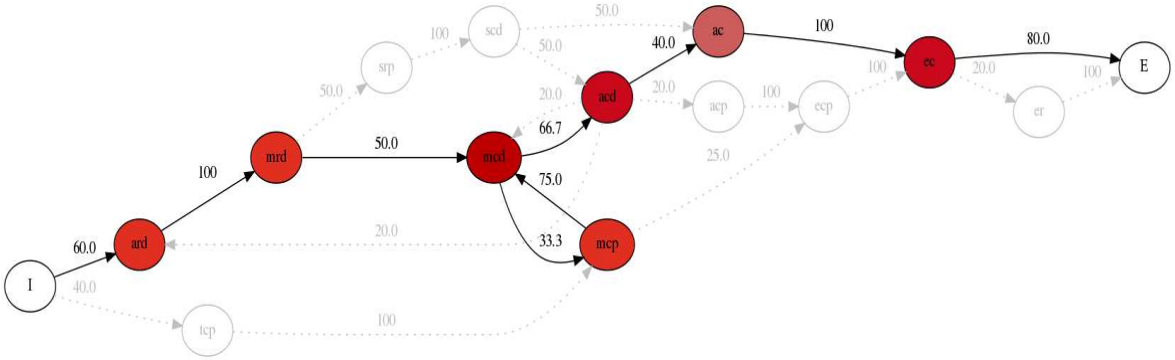

Figura 3.20 - Resultado da aplicação da terceira e última restrição.

Se forem quebradas todas as ligações entre o inicio e o fim das várias sessões de consulta sobre o cubo e declaradas na Cadeia de Markov colorida correspondente (o que indica que durante as sessões OLAP não foi revelada qualquer incidência considerada forte sobre um conjunto específico de vistas). Assim, o conjunto de vistas indicadas pelo método de selecção será apenas composto pelas vistas coloridas na primeira fase do método, na qual, como já foi referido, realçaram-se as vistas tendo em conta as quatro características estipuladas - intervalo de tempo em análise, dimensões de análise, limite mínimo de utilização e o limite máximo de selecção. Caso se verificasse tal situação, seriam escolhidas apenas as vistas representadas na figura 3.15.

Desta forma, para o caso de estudo seguido nesta secção (definido na figura figura 3.8), a *lattice* que deveria ser materializada no servidor OLAP encontra-se representado na figura 3.21. Nesta figura podemos ver as sete vistas coloridas dentro de um conjunto total de cinquenta e quatro vistas.

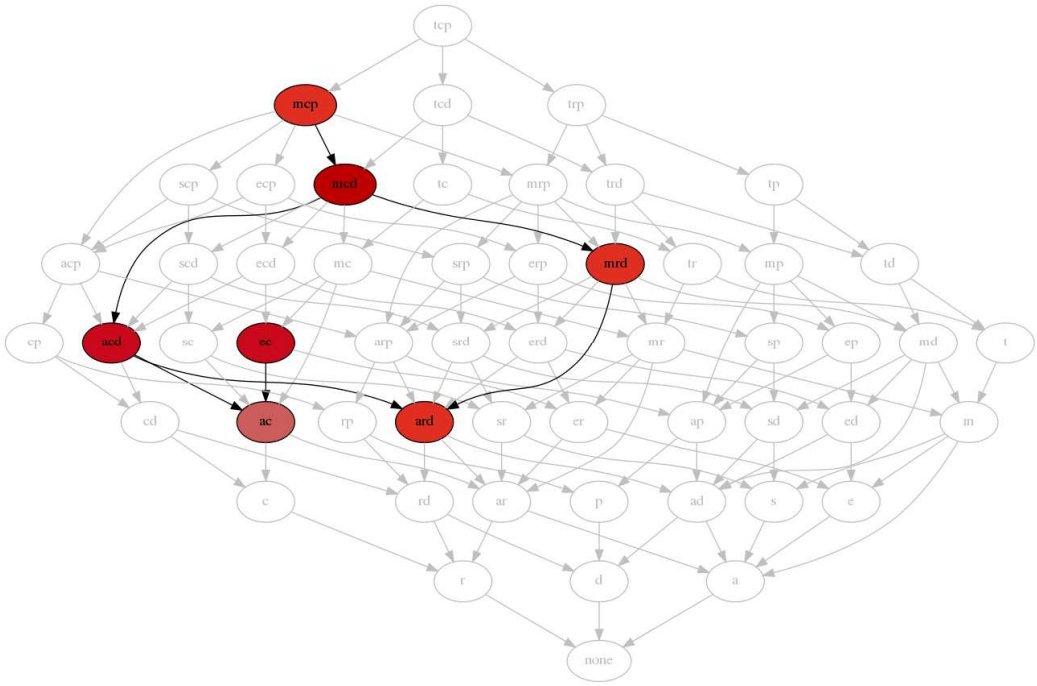

Figura 3.21 - Ilustração do resultado obtido com a execução do método M3.

Capítulo 4

Validação do Método Desenvolvido

Tal como referiu o reconhecidíssimo estatístico Dr. William Edwards Deming, "You can't manage what you don't measure", ou seja, como transparece este ditado usado nas áreas relacionadas com a gestão, a menos que se consiga medir algo, nunca se terá a absoluta certeza se poderá ser considerado melhor, ou pior no momento que se tenta efectuar uma comparação. Desta forma, ao elaborar o método de selecção ao longo dos trabalhos de dissertação realizados, definiu-se como um dos objectivos fulcrais efectuar vários testes comparativos com outros algoritmos de selecção, para que desta forma se consiga obter uma noção dos desempenhos conseguidos pelo método M3 [I. Molyneaux 2009]. De referir que o método M3 revelou-se bastante efectivo nos testes laboratoriais desenvolvidos para validação da sua execução e prova da sua viabilidade. Mas para sustentar devidamente esta conclusão, necessita-se de comparar o seu desempenho e resultados com outros métodos similares, com objectivos idênticos, num cenário típico de aplicação real.

Para que se obtenha a desejada clarificação, em termos de desempenho conseguidos pelo método M3, foram efectuados três tipos de testes em dois cenários diferentes, ou por outras palavras, sobre dois esquemas dimensionais diferentes. Os três testes efectuados pretendem experimentar o método M3 a nível do seu desempenho, enquanto processo de selecção de vistas, como também medir a qualidade do conjunto de vistas seleccionadas. A par dos testes efectuados sobre o M3, realizaram-se também os mesmos testes sobre os dois esquemas dimensionais referidos utilizando outros dois algoritmos de selecção. Desta forma, conseguiu-se obter uma base de comparação em que se pôde observar a performance do método M3 relativamente com outros métodos da mesma linha [WWW07].

Assim, neste capítulo, começa-se por abordar os testes realizados, de forma a entender as características que cada um dos testes pretende verificar e provar. De seguida, ilustra-se os resultados obtidos com a execução de cada um dos testes, sendo elaboradas, para cada um deles,

as observações mais pertinentes com o intuito de ajudar o leitor na análise dos valores constatados.

4.1 Planeamento dos Testes de Desempenho

Para que os testes de desempenho demonstrem com efectividade as características do método M3, foi planeada a execução de três tipos de testes. A par do método M3, definiu-se também a execução dos algoritmos PBS [Shuklaet al. 1998] e o HRU [Hanusseet al. 2009], para as mesmas condições de teste. Para que não ficasse a ideia de que os resultados obtidos estavam muito dependentes do esquema dimensional utilizado, realizaram-se os três testes referidos em dois esquemas dimensionais diferentes. Desta forma criou-se uma base de comparação, em que se podem constar e comparar valores de execução do método M3, com um algoritmo pouco recente (o algoritmo PBS que foi apresentado em 1998) e outro e bastante recente (o algoritmo HRU que foi apresentado em 2009).

Para que os testes de desempenho identificassem de forma esclarecedora a performance do método implementado, ficou definido que seriam efectuadas várias provas, tanto em termos de recursos despendidos como da qualidade das vistas seleccionadas. Desta forma foram realizadas os seguintes testes:

- **Tempo de execução** - De forma a perceber o tempo de execução despendido por cada um dos métodos de selecção utilizados nos testes comparativos, efectuou-se a medição dos tempos de processamento (em milissegundos) de cada um dos métodos.

- **Memória efectivamente ocupada** - Os métodos M3, HRU e PBS possuem um parâmetro de entrada em comum, que é o limite máximo de espaço a poder ser despendido no servidor OLAP. Muito embora esse limite nunca seja ultrapassado, existem abordagens que tentam alcançar ao máximo esse limite, enquanto outras deixam uma folga, por vezes considerável, de espaço que poderia ser aproveitado. Neste teste pretendeu-se observar a efectividade de cada um dos métodos consoante o montante de memória efectivamente ocupado em detrimento do limite máximo estipulado.

- **Qualidade do conjunto de vistas seleccionado** - De forma a perceber o contributo que cada um dos métodos de selecção pode desenvolver, ao indicar o conjunto de vista óptimo a materializar (consoante os pressupostos que cada um utiliza), foram efectuadas várias simulações de *queries* MDX sobre um servidor OLAP. Foram simuladas 300 submissões e registado o número de vezes em que houve a necessidade de consultar o *data warehouse* para atender à consulta efectuada, bem como o número de vezes em que o servidor OLAP já possuía materializado o resultado pretendido. As *queries* MDX foram lançadas aleatoriamente, mas de forma a dar prioridade às consultas mais frequentemente efectuadas nas sessões realizadas antes da execução dos métodos de selecção.

Desta forma, definiu-se um ambiente em que se pode comparar o método M3, tendo em conta os aspectos principais que se pretendem aperfeiçoar num método de selecção. Pretendia-se sempre que o método fizesse a computação do cubo *iceberg* de forma rápida, que aproveitasse ao máximo a quantidade de recursos disponibilizados (sempre que necessário) e por fim, que indicasse um conjunto de vistas que evitasse a constante sobrecarga de consultas sobre o *data warehouse,* sendo que isso poderia implicar uma demora substantivamente maior, do que aquela quando já se possuísse armazenado os resultados no servidor OLAP.

4.1.1 Definição dos Testes Executados

Para concretizar os testes pretendidos sobre o método M3, definiram-se dois esquemas dimensionais distintos. Os dois esquemas dimensionais de facto parecem ser muito parecidos, dado que possuem quase os mesmos números de *cuboids*, mas ambos serviram para testar duas situações bem diferentes. Em primeiro lugar definiu-se um esquema dimensional e um limite máximo de espaço a ser utilizado, em que o método M3 (de forma folgada) não necessita de atingir o limite máximo de selecção. Nesta primeira situação, os *cuboids* que sofreram uma maior incidência cabem perfeitamente no espaço estipulado. De forma contrária, em segundo lugar, definiu-se um outro esquema dimensional com um limite de espaço muito curto. Desta forma, criou-se uma situação em que se pode testar a capacidade de resumo do algoritmo de selecção na escolha do conjunto de vistas. Assim, a primeira situação irá ser denominada de "Esquema 1" para efeitos de referências futuras e a segunda fica referenciada como "Esquema 2".

Características do Esquema 1

O primeiro esquema dimensional encontra-se apresentado na figura 4.1 e a respectiva *lattice* na figura 4.2. Nesta última podemos ver quais os pesos que foram atribuídos a cada *cuboid*. Os pesos que se atribuíram a cada um dos 32 *cuboids* utilizados e o limite de espaço estipulado, criaram um caso propício para demonstrar a efectividade do método M3, numa situação em que se pode constatar a forma como ele consegue pôr de parte as vistas que não sofre consultas de forma frequente.

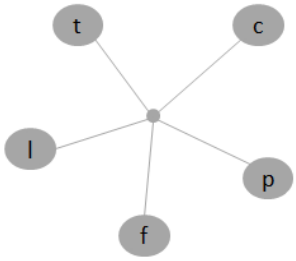

Figura 4.1 - Esquema dimensional definido para o primeiro conjunto de testes.

A *lattice* (representada na figura 4.2) possui no total um peso de 12376 MB, sendo o limite máximo do espaço definido para a selecção de 5700 MB, o que nos garante uma redução de pelo menos 46% do espaço necessário para a sua materialização no servidor OLAP.

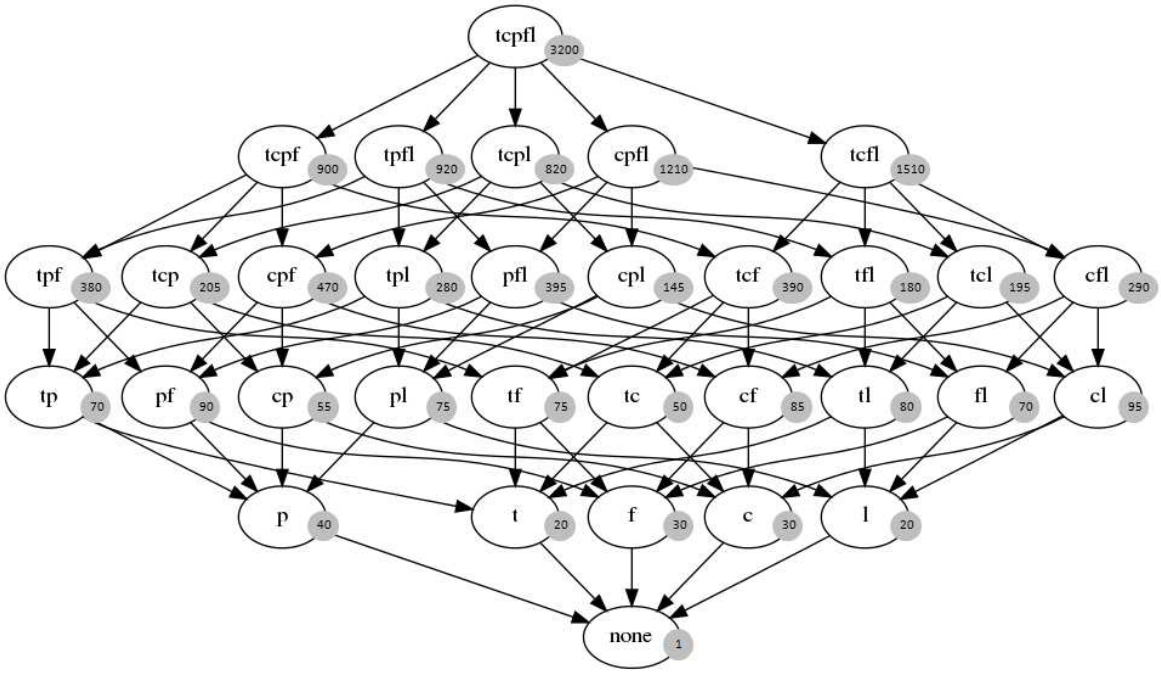

Figura 4.2 - *Lattice* do esquema dimensional representado na figura 4.1.

Características do Esquema 2

O esquema dimensional referente ao segundo esquema de testes, encontra-se definido na figura 4.3 e a respectiva *lattice* na figura 4.4. Os pesos que se atribuíram a cada um dos 27 *cuboids* e o limite máximo de espaço estipulado como parâmetro, definem um caso em que se pode demonstrar a capacidade do método M3 em situações para as quais não se possui qualquer folga de espaço para efectuar uma perfeita escolha dos *cuboids* mais consultados, o que permite testar, assim, a sua capacidade de aproveitamento do espaço proporcionado.

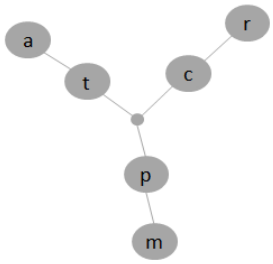

Figura 4.3 - Esquema dimensional definido para o segundo conjunto de testes.

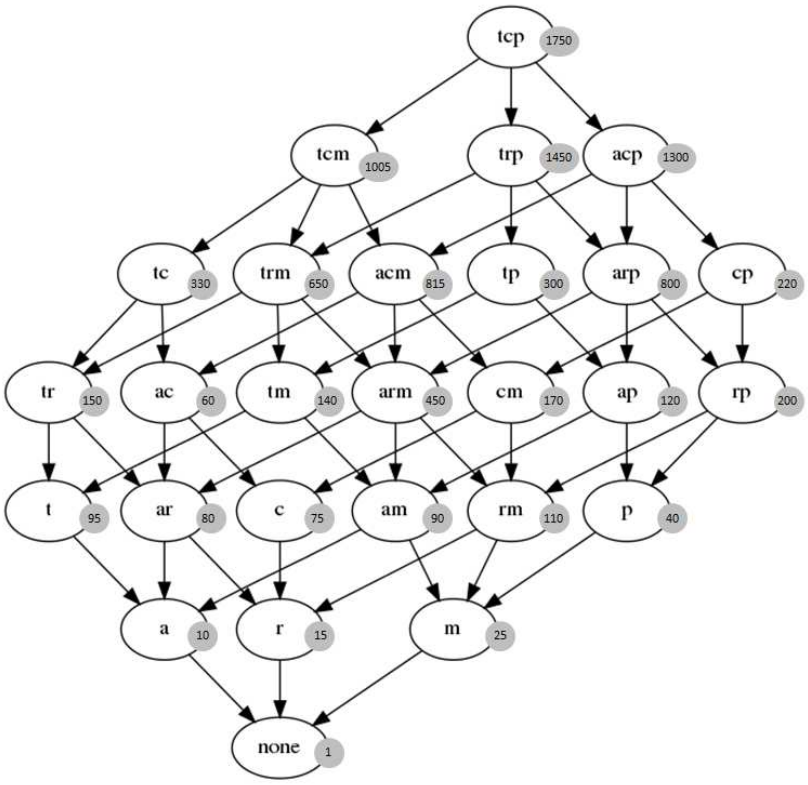

Figura 4.4 - *Lattice* corresponde ao esquema dimensional representado na figura 4.3.

A *lattice* possui no total um peso de 10451 MB, sendo o limite máximo de espaço definido para a selecção de 4200 MB, o que nos pode garantir uma redução de pelo menos 40% do espaço necessário para a sua materialização no servidor OLAP.

4.1.2 Algoritmos Utilizados na Comparação

Para se efectuarem as comparações com outros algoritmos de selecção, foram escolhidos os algoritmos PBS [Shuklaet al. 1998] e HRU [Hanusseet al. 2009] devido ao facto de ambos serem bastante conhecidos e pelo facto de um ter deles sido um dos primeiros algoritmos avançados pela comunidade científica (o PBS), ao invés do outro, que foi apresentado num passado bastante recente (o HRU). Recordando os dois algoritmos de selecção de forma resumida, cada um pode ser caracterizado da seguinte forma:

- **PBS** - A selecção baseia-se no espaço máximo que as vistas seleccionadas poderão ocupar. A vista correspondente à tabela de factos é sempre materializada. O conjunto de vistas seleccionado é constituído pelas vistas que têm a menor densidade de dados possível. Este algoritmo caracteriza-se também por avançar as vistas indicadas para materialização com baixos tempos de processamento.

- **HRU** - A selecção baseia-se no espaço máximo que as vistas seleccionadas poderão ocupar. A vista correspondente à tabela de factos é sempre materializada e o conjunto de vistas devolvido pelo algoritmo é constituído pelas vistas que, em termos absolutos, permitem maximizar o benefício (relativamente ao espaço ocupado), tendo em conta o conjunto total seleccionado. Uma das desvantagens deste algoritmo de selecção é o de possuir longos tempos de execução para esquemas dimensionais com muitas dimensões.

Definiu-se, então, um ambiente em que facilmente se concretiza a comparação evolutiva que o método de selecção M3 atinge. Os dois algoritmos seleccionados efectuam uma selecção de forma genérica, sem ter em conta qualquer perfil de preferência ou probabilidade de consultas futuras. Também podemos testar a situação em que se constatam os desempenhos de dois grupos diferentes de métodos de selecção, uma vez que o método M3 tem como principal objectivo o de indicar um conjunto de vistas, para as quais existe um elevado grau de probabilidade de serem consultadas logo de seguida.

4.2 Testes e Resultados

Após definidos os conjuntos de testes a executar, com o objectivo de perceber o desempenho do método M3, procedeu-se à execução dos testes planeados. Com a execução de cada um dos três métodos de selecção sobre cada um dos esquemas dimensionais, foram anotados os seus desempenhos e as vistas que foram seleccionadas. No fim, elaborou-se um resumo dos valores obtidos para que fosse possível efectuar a comparação dos desempenhos obtidos em cada um dos testes Realizados. De referir, contudo, algumas curiosidades interessantes, relativamente aos testes efectuados:

- Os métodos de selecção M3, HRU e PBS foram todos implementados em Java - versão "Java SE Runtime Environment 1.6.0-24".
- O sistema operativo no qual foram executados os testes foi o Fedora Release 14 (Laughlin).
- A máquina em que foram executados os testes tinha um processador de 32 bits, Intel Centrino Duo (1.73 GHz, *FSB* de 533 MHz e *cache* de 2 MB) e 2GB de RAM (DDR2-533 MHz).

4.2.1 Execução e Análise dos Teste de Desempenho

A execução dos testes dividiu-se estruturalmente em duas partes, sendo que na primeira executou-se os testes sobre o Esquema 1 e na segunda parte sobre o Esquema 2. Em cada um dos testes, começou-se por apresentar os valores obtidos, executando-se em primeiro lugar o algoritmo PBS, de seguida o algoritmo HRU e, por fim, o método M3. Para cada um dos casos analisados apresentam-se de seguida os valores obtidos relativamente ao tempo de execução, memória efectivamente ocupada e qualidade do conjunto vistas seleccionadas.

Execução dos Teste Sobre o Esquema 1

Após executados os testes efectuados sobre o Esquema 1, recorrendo ao algoritmo de selecção PBS, obteve-se uma selecção composta por 23 vistas, representadas a cinzento na figura 4.5. As restantes 9 vistas, representadas a branco, foram as vistas que foram excluídas do conjunto de selecção. A não ser o *cuboid root*, todos os outros *cuboids* que foram seleccionados possuem o menor peso possível. Por isso, o conjunto de vistas seleccionado encontra-se, sobretudo, na base da *lattice*.

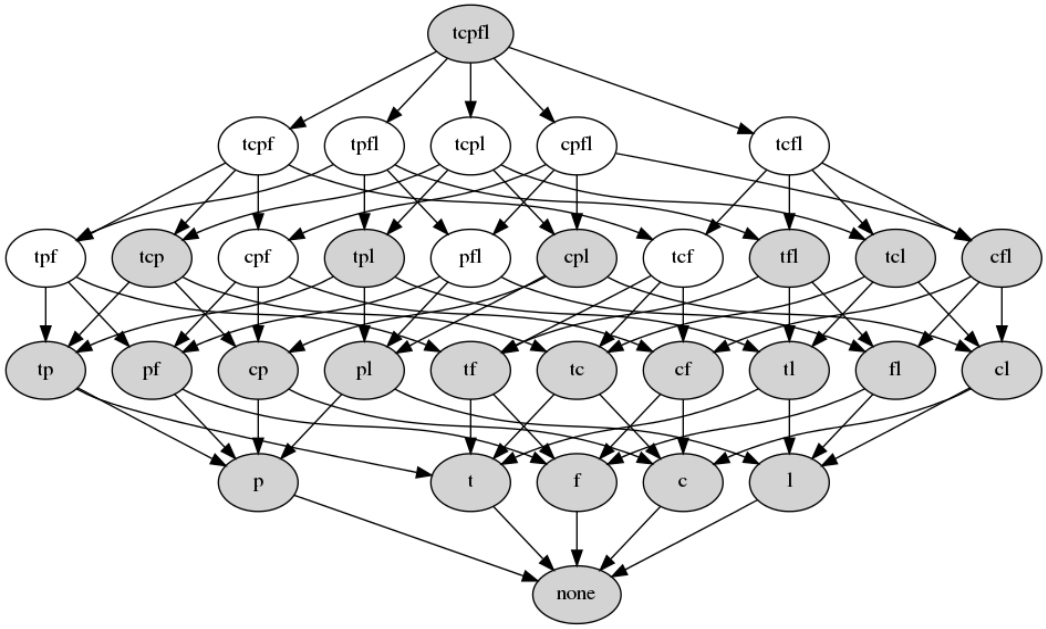

Figura 4.5 - Cubo *iceberg* produzido pelo algoritmo PBS com base no Esquema 1.

Os resultados obtidos para cada um dos testes efectuados encontram-se descritos na tabela 4.1.

Tempo de execução	3 milissegundos
Espaço efectivamente ocupado	5381 MB em 5700 MB possíveis
Qualidade do conjunto de vistas seleccionado	Houve a necessidade de efectuar 124 consultas sobre o *data warehouse*

Tabela 4.1 - Resultados da execução do algoritmo PBS sobre o Esquema 1.

Com a execução do algoritmo HRU sobre o Esquema 1, obteve-se uma selecção de 18 vistas, que estão representadas a cinzento na figura 4.6. As restantes 14 vistas, representadas a branco foram as que não foram incluídas no conjunto de selecção. Como se pode verificar na figura 4.6, o *cuboid*

root, à semelhança do algoritmo PBS, também é materializado, mas as restantes vistas seleccionadas apresentaram uma distribuição mais dispersa, tendo como principal característica o facto de que os *cuboids* seleccionados maximizam o benefício, relativamente ao espaço que ocupam no conjunto seleccionado.

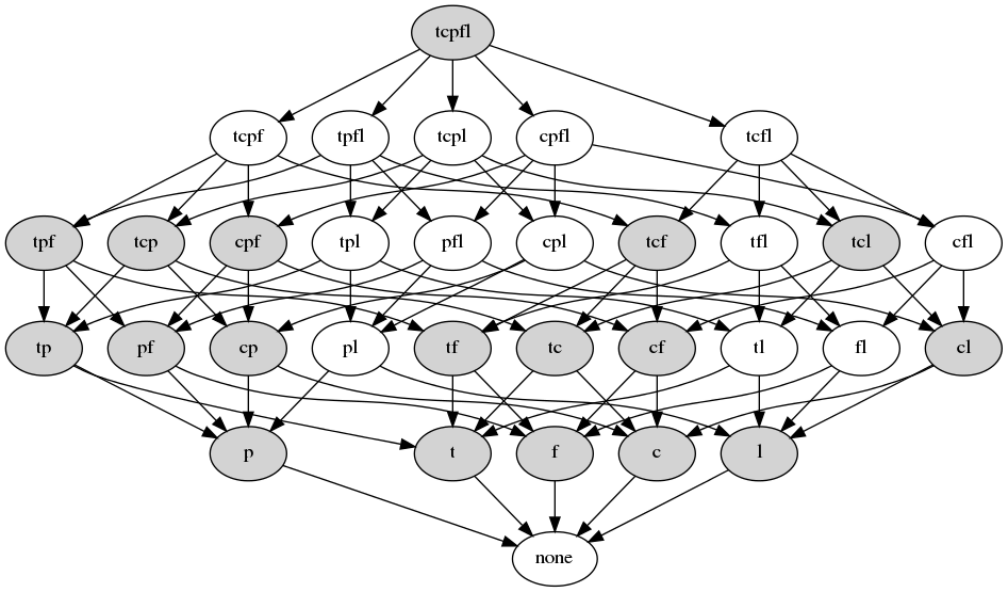

Figura 4.6 - Cubo *iceberg* produzido pelo algoritmo HRU para o Esquema 1.

Os resultados obtidos relativos aos três testes efectuados foram os seguintes:

Tempo de execução	33 milissegundos
Espaço efectivamente ocupado	5500 MB em 5700 MB possíveis
Qualidade do conjunto de vistas seleccionado	Houve a necessidade de efectuar 108 consultas sobre o *data warehouse*

Tabela 4.2 - Resultados da execução do HRU sobre o Esquema 1.

Finalmente sobre o Esquema 1, executou-se o método de selecção M3. Com este método obteve-se um conjunto de selecção constituído por 13 vistas, que se encontram representadas a cinzento na figura 4.7. A branco encontram-se as 19 vistas que foram excluídas do conjunto final de selecção. O cubo *iceberg* é composto pelas vistas que maior probabilidade têm de serem consultadas em termos futuros. Pelo conjunto de vistas seleccionado, pode-se observar que existem grandes diferenças relativamente aos conjuntos devolvidos pelos algoritmos PBS e HRU. Isso é devido ao facto de que estes algoritmos terem pressupostos de selecção diferentes do M3. O número de vistas seleccionado pelo M3 é inferior aos restantes dois algoritmos, o que demonstra a sua capacidade de resumo e discriminação de vistas que não possuem frequência de consulta. Os resultados relativos a cada um dos testes efectuados encontram-se apresentados na tabela 4.3.

Tempo de execução	106 milissegundos
Espaço efectivamente ocupado	3220 MB em 5700 MB possíveis
Qualidade do conjunto de vistas seleccionado	Houve a necessidade de efectuar 62 consultas sobre o *data warehouse*

Tabela 4.3 - Resultados da execução do algoritmo M3 sobre o Esquema 1.

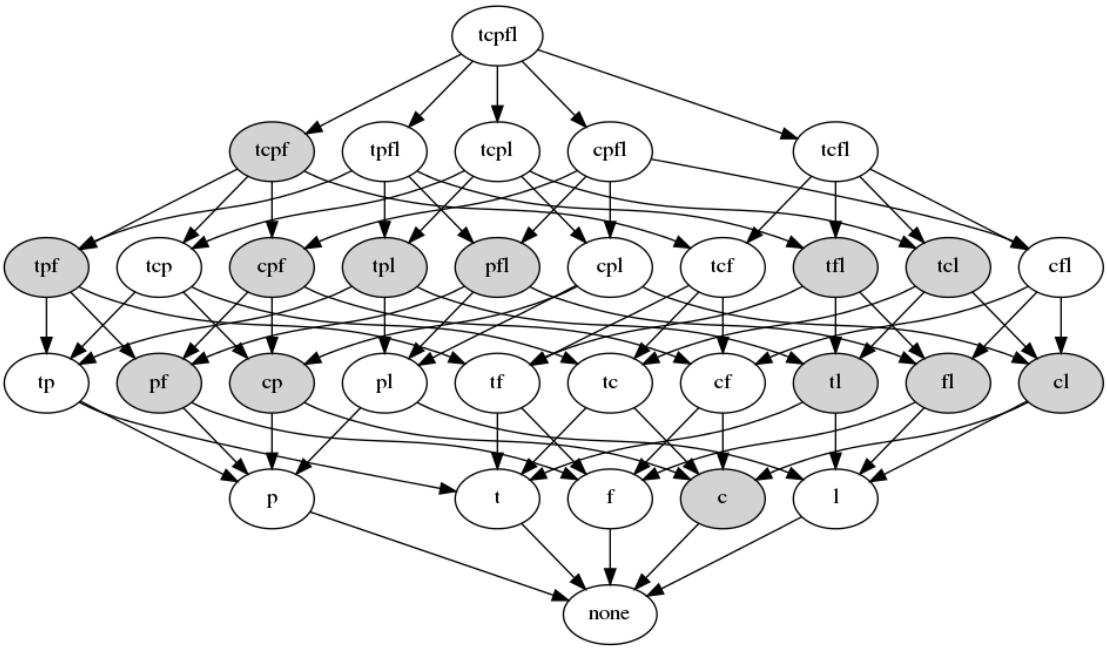

Figura 4.7 - Cubo *iceberg* produzido pelo algoritmo M3 para o Esquema 1.

Execução dos Teste Sobre o Esquema 2

Executados os testes com o algoritmo PBS sobre o Esquema 2, ficaram seleccionadas 20 vistas, que estão representadas a cinzento na figura 4.8. As restantes 7 vistas, representadas a branco, foram excluídas do conjunto inicial de vistas. Como se pode confirmar novamente pela *lattice* apresentada anteriormente (na figura 4.8), o PBS escolheu essencialmente as vistas provenientes da base, uma vez que estas são as que usualmente ocupam menor espaço, à excepção do *cuboid root* que, como sabemos, é sempre materializado. Os resultados relativos a cada um dos testes efectuados encontram-se apresentados na tabela 4.4.

Tempo de execução	3 milissegundos
Espaço efectivamente ocupado	3981 MB em 4200 MB possíveis
Qualidade do conjunto de vistas seleccionado	Houve a necessidade de efectuar 169 consultas sobre o *data warehouse*

Tabela 4.4 - Resultados da execução do PBS sobre o Esquema 2.

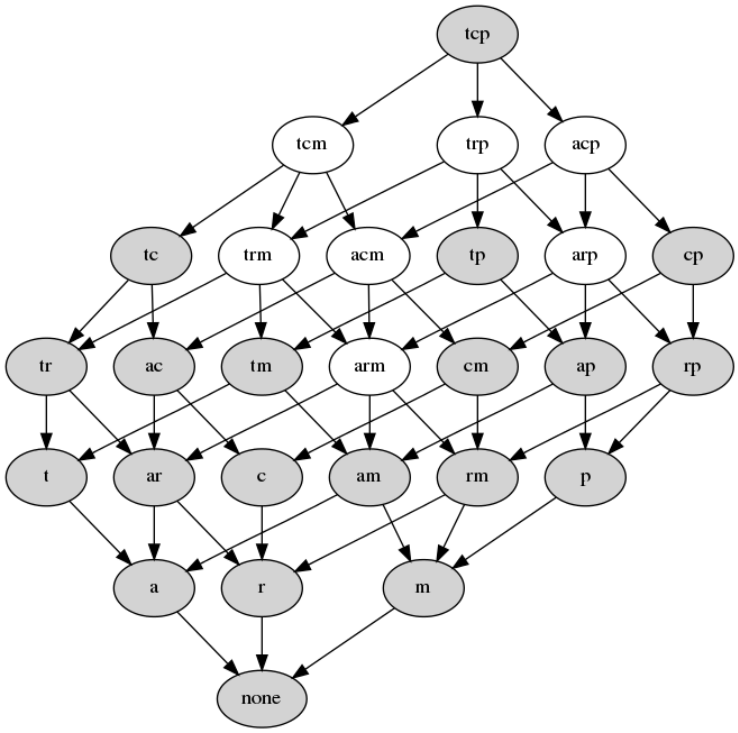

Figura 4.8 - Cubo iceberg produzido pelo algoritmo PBS para o Esquema 2.

Após efectuados e analisados os testes com o algoritmo HRU sobre o Esquema 2, obteve-se uma selecção composta por 13 vistas. Na figura 4.9 representam-se a cinzento as vistas escolhidas. Neste teste 14 vistas (representadas a branco) ficaram fora do conjunto de selecção. Como se pode verificar na figura 4.9, o conjunto de vistas indicado pelo HRU foi bem diferente do indicado pelo PBS, no qual a percentagem total de vistas seleccionadas foi substancialmente inferior, tendo novamente sido seleccionadas, unicamente, as vistas situadas principalmente na base da *lattice*. Os resultados relativos a cada um dos testes efectuados encontram-se apresentados na tabela 4.5.

Tempo de execução	28 milissegundos
Espaço efectivamente ocupado	4160 MB em 4200 MB possíveis
Qualidade do conjunto de vistas seleccionado	Houve a necessidade de efectuar 208 consultas sobre o *data warehouse*

Tabela 4.5 - Resultados da execução do HRU sobre o Esquema 2.

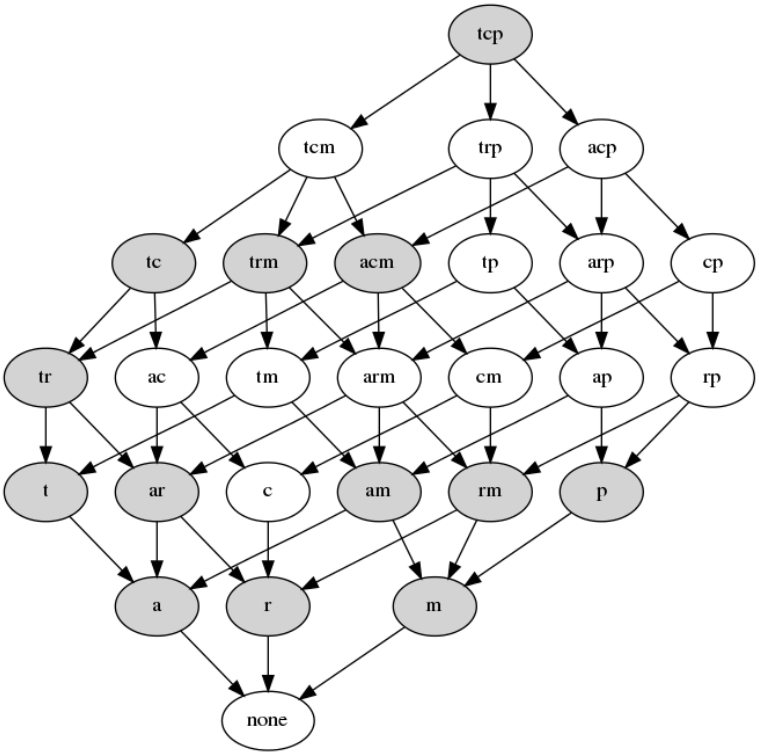

Figura 4.9 - Cubo iceberg produzido pelo algoritmo HRU para o Esquema 2.

Por fim, testou-se o método de selecção M3 sobre o Esquema 2. Neste teste obteve-se uma selecção de 10 vistas, tal como se pode ver representadas a cinzento na figura 4.10. As restantes 17 vistas, foram excluídas do conjunto de selecção. Mais uma vez o conjunto de vistas seleccionado pelo método M3 é consideravelmente diferente dos outros dois algoritmos (figuras 4.10, 4.9 e 4.8). De referir que nem todas a vistas com grande probabilidade de consulta foram seleccionadas, devido ao facto que o limite máximo de espaço atribuído ser pequeno. Assim, testámos o M3 de forma intensiva naquilo que dizia respeito à sua capacidade de indicar as vistas mais consultadas. Os valores obtidos em cada um dos três tipos de testes encontram-se representados na tabela 4.6.

Tempo de execução	28 milissegundos
Espaço efectivamente ocupado	4075 MB em 4200 MB possíveis
Qualidade do conjunto de vistas seleccionado	Houve a necessidade de efectuar 102 consultas sobre o *data warehouse*

Tabela 4.6 - Resultados da execução do método M3 sobre o Esquema 2.

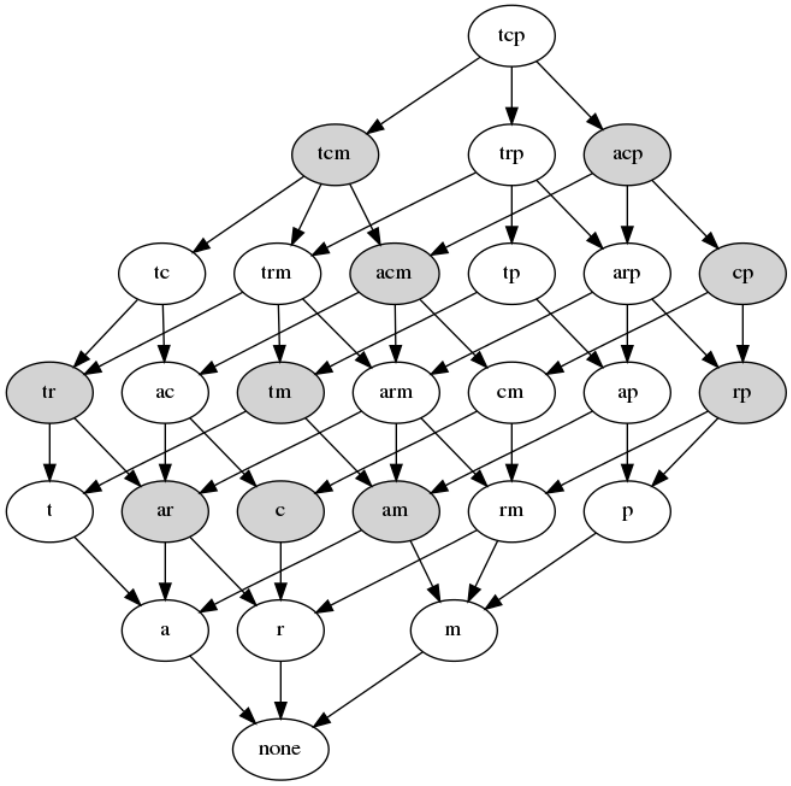

Figura 4.10 - Cubo iceberg produzido pelo algoritmo M3 para o Esquema 2.

4.2.2 Análise Geral dos Resultados Obtidos

Após a recolha dos valores registados, elaboraram-se os gráficos onde se pode comparar os tempos de execução, o montante de memória efectivamente ocupados e a taxa de sucesso após simuladas consultadas sobre um servidor OLAP, dividindo o processo pela execução dos dois esquemas dimensionais definidos. Pretende-se assim facilitar a comparação dos desempenhos observados, com o recurso a gráficos mas também com a respectiva descrição e interpretação dos mesmos, obtendo assim a devida compreensão dos desempenhos observados pelo método M3.

Análise dos Testes Relativos ao Esquema 1

Na figura 4.11 encontram-se apresentados os tempos de execução dos algoritmos de selecção PBS, HRU e M3, relativamente aos testes de desempenho efectuados sobre o Esquema 1. Como se pode observar, o M3 é consideravelmente mais demorado. De facto, já se esperava que o tempo de execução deste algoritmo atingisse esta diferença temporal, dado que, se observarmos a organização do M3, verificamos que o número de processos despoletados é muito maior. Assim, este algoritmo não deve ser utilizado em situações nas quais se pretende obter conjuntos de vistas com tempos de execução realmente baixos.

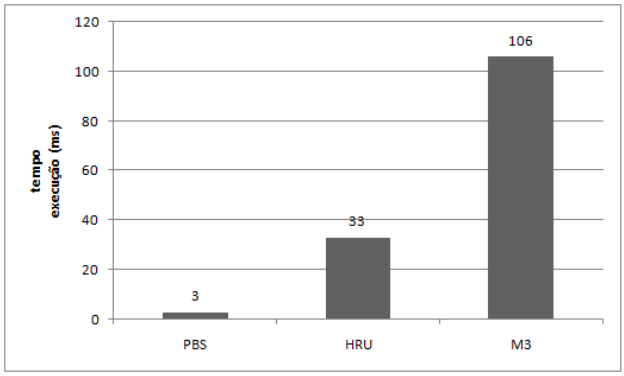

Figura 4.11 - Tempos de execução dos testes efectuados sobre o Esquema 1.

A quantidade de memória que foi gasta com a execução dos algoritmos PBS, HRU e M3, sobre o esquema dimensional correspondente ao Esquema 1, pode ser observada na figura 4.12. Recorde-se que o limite máximo de memória estipulado para cada teste foi de 5700 MB. Como se pode constatar, os algoritmos PBS e HRU usaram uma maior parcela do espaço disponível, enquanto que o M3 utilizou apenas cerca de 56% do espaço disponível. Isto deve-se ao facto do número de vistas frequentemente consultadas durante as sessões OLAP terem sido muito restritas. Deste modo o M3 conseguiu efectuar um resumo efectivo dos conteúdos realmente mais pesquisados e que mais interessam aos agentes de decisão. Para comprovar esta afirmação, na figura 4.13 está apresentada a taxa de sucesso da escolha efectuada pelo M3.

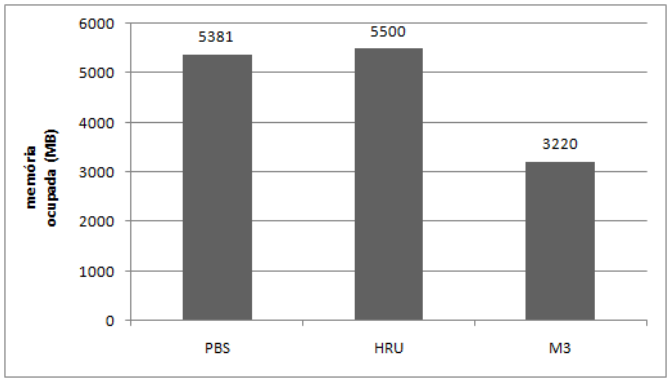

Figura 4.12 - Memória utilizada nos testes efectuados sobre o Esquema 1.

De seguida, na figura 4.13, podemos ver um gráfico que revela o número de vezes que o servidor OLAP conseguiu dar resposta às consultas recebidas sem recorrer a uma pesquisa sobre o *data warehouse* (situação representada a azul) e o número de vezes em que não possuía a vista materializada no servidor e se viu obrigado a desenvolver uma pesquisa sobre o *data warehouse* (situação representada a vermelho).

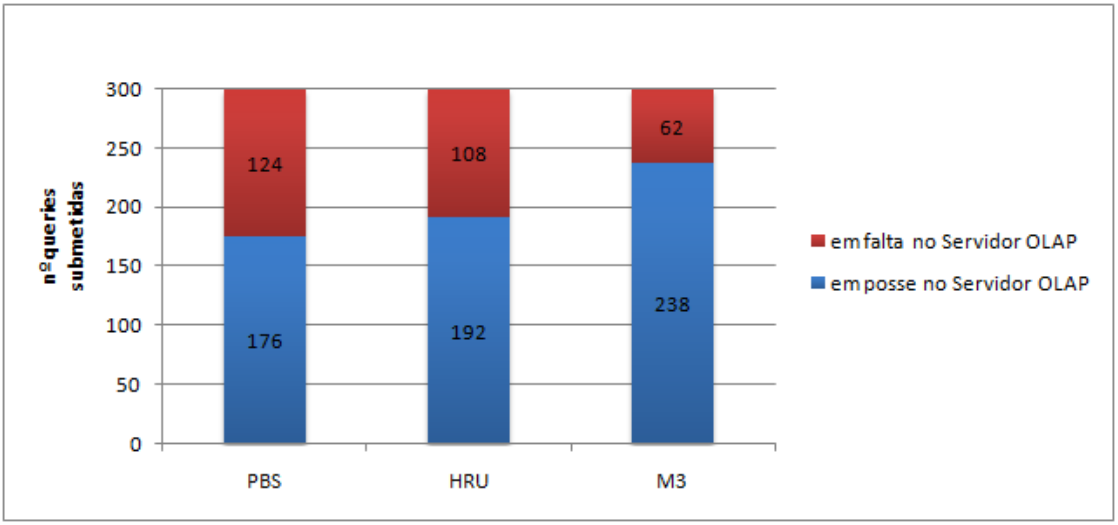

Figura 4.13 - Qualidade do conjunto de vistas seleccionado sobre o Esquema 1.

Com base nos resultados obtidos, podemos dizer que o desempenho do algoritmo M3 nos testes efectuados com recurso ao Esquema 1 é que, de facto, o seu tempo de execução é prolongado, mas a qualidade do conjunto de vistas indicado pelo método é significativamente superior. Conseguiu-se, assim, fazer uma poupança significativa do espaço total armazenado, observando-se uma taxa bastante boa de respostas a consultas sem haver necessidade de pesquisas complementares sobre o *data warehouse*. Desta forma, fica demonstrado que os pressupostos assumidos para o M3 providenciam bons resultados, quando comparados com os algoritmos de selecção utilizados nesta dissertação.

Análise dos Testes Relativos ao Esquema 2

Os tempos de execução obtidos para os algoritmos PBS, HRU e M3, relativamente aos testes de desempenho efectuados sobre o Esquema 2, podem ser observados na figura 4.14. Mais uma vez, pode-se constatar que o M3 é de facto mais demorado, situação facilmente verificável nos tempos de espera relativos com a demora de execução. Segunda a literatura da especialidade, existem diversos algoritmos de selecção com valores consideravelmente melhores que o M3. Nos artigos em que se apresentam os algoritmos PBS [Shuklaet al. 1998] e HRU [Hanusseet al. 2009], pode-se verificar que um dos aspectos que ambos os algoritmos procuram alcançar são tempos de execução reduzidos. De facto, o método M3 não pretende minimizar tempos de construção de

cubos *iceberg*, mas sim minimizar os tempos de espera das consultas efectuadas sobre o servidor OLAP, seleccionando as vistas com maior probabilidade de consulta.

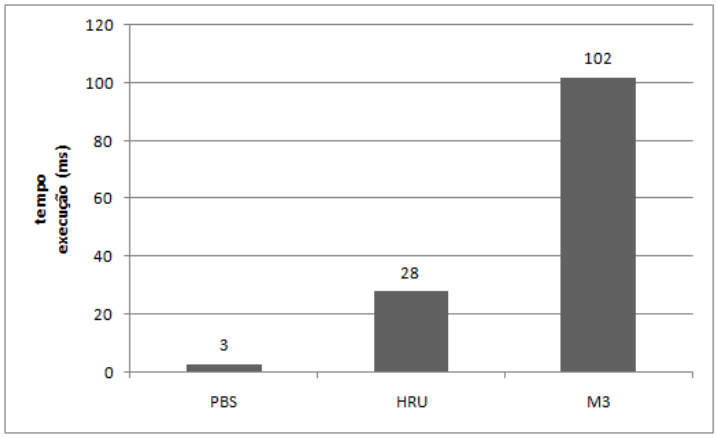

Figura 4.14 - Tempos de execução dos testes efectuados sobre o Esquema 2.

Na figura 4.15, estão identificados os valores de memória realmente utilizados durante a execução dos algoritmos PBS, HRU e M3 sobre o esquema dimensional correspondente ao Esquema 2. O limite máximo de memória inicialmente definido foi de 4200 MB. Verifica-se que ambos os métodos de selecção tentaram preencher ao máximo o espaço disponibilizado. Isto deve-se ao facto do limite especificado ser de facto pequeno quando comparado com o espaço ocupado pela *lattice* no total. Desta forma, pode-se verificar que no caso especificado no Esquema 2, o método M3 necessitaria de um limite de espaço com um valor superior para concretizar em pleno os seus pressupostos de selecção.

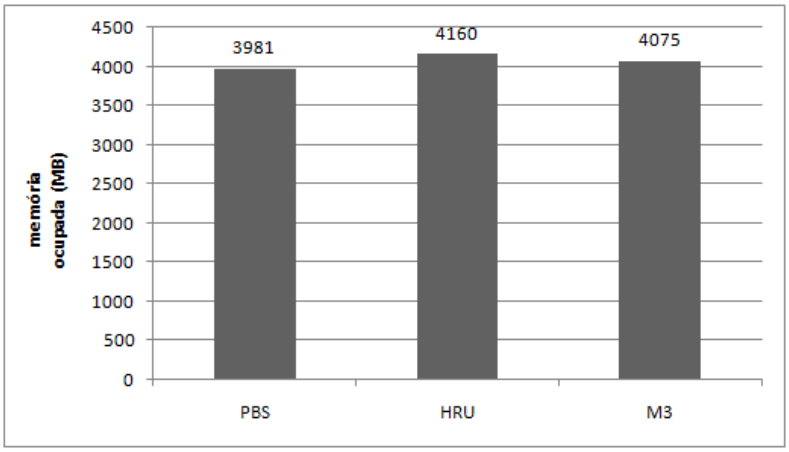

Figura 4.15 - Montantes de memória gastos nos testes efectuados sobre o Esquema 2.

Na figura 4.16 está apresentado um gráfico em que se pode observar como correram as diversas simulações efectuadas sobre cada conjunto de vistas seleccionado, com o objectivo de avaliar a qualidade dos resultados que cada um dos métodos de selecção produziu. A azul representa-se o

número de vezes que o servidor OLAP conseguiu responder às *queries* recebidas sem consultar o *data warehouse*, enquanto que a vermelho representa-se o número de vezes que o servidor OLAP teve mesmo necessidade de recorrer ao *data warehouse* para responder às consultas efectuadas.

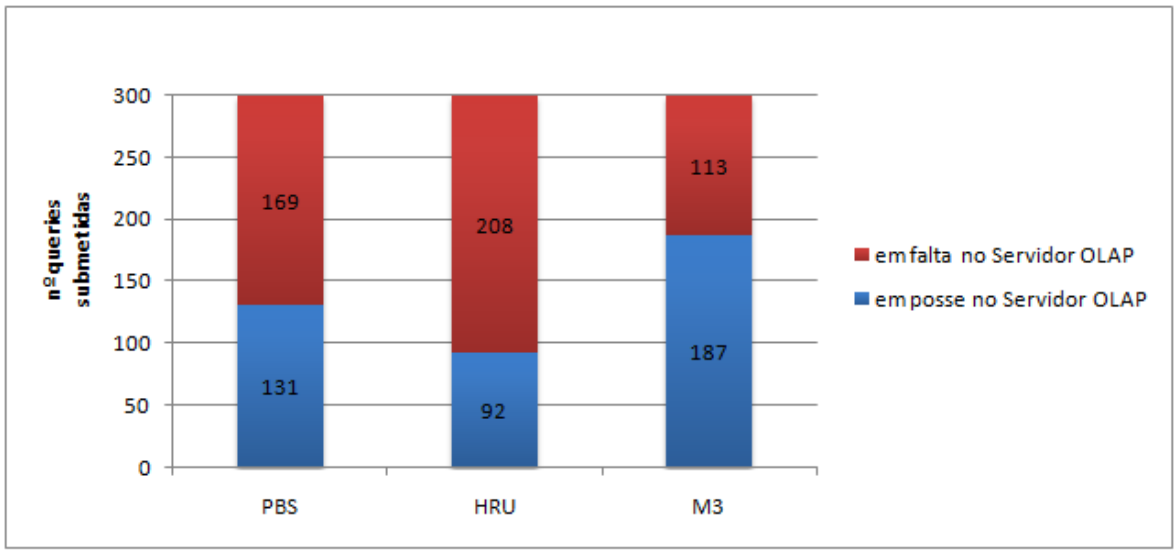

Figura 4.16 - Qualidade do conjunto de vistas seleccionado sobre o Esquema 2.

Resumindo, as principais conclusões que se podem retirar, relativamente ao desempenho demonstrado pelo M3 durante os testes realizados sobre o Esquema 2, são semelhantes às observações retiradas anteriormente com a execução do conjunto de testes executados sobre o Esquema 1. Tal como se pode verificar na figura 4.16, pelo facto de serem seleccionadas as vistas mais consultadas ao longo das sessões OLAP, obtém-se uma maior eficiência na resposta das *queries* submetidas no servidor, uma vez que são reduzidos o número de consultas que se necessita efectuar sobre o *data warehouse*. Mais uma vez, constatou-se que o tempo de execução do M3 é significativamente superior, quando comparados com os outros algoritmos de selecção.

O esquema definido neste conjunto de testes possibilitou retirar mais uma conclusão importante: o método M3 comprovou as suas capacidades de identificar um conjunto de vistas o mais resumido possível (quando o limite atribuído ao espaço máximo torna-se curto), confirmando uma vez mais que os pressupostos que definem as selecções efectuadas pelo método M3, são de facto vantajosos.

Capítulo 5

Conclusões e Trabalho Futuro

5.1 Conclusões Finais

Tem sido discutido pelos vários investigadores da área, de forma muito intensa, o problema da escolha de um cubo *iceberg* que se identifique como sendo o mais adequado para ser materializado no processo de implementação de um cubo OLAP [WWW04]. Muitos esforços e soluções podem-se nesta altura referir. O trabalho nesta vertente promete ser contínuo, dado que o desenvolvimento e a utilização das ferramentas analíticas de dados têm verificado grande adesão nos últimos anos. Ao longo da última década, tem-se verificado que as empresas e os grupos económicos optam cada vez mais pela implementação de sistemas de *data warehousing* como um meio privilegiado de organização do seu património de dados, que tende a ser maior de dia para dia [WWW05]. Sobre este tipo de sistema são aplicados, de forma quase implícita, ferramentas analíticas de dados, porque se revelam como sendo as mais propícias para efectuar a exploração dos dados mantidos nos *data warehouses*. À medida que as ferramentas analíticas de dados ganham expressão, como forma de proporcionar aos agentes de decisão um meio com enormes vantagens para os processos de tomada de decisão, cada vez mais se tenta encontrar formas que melhoram a performance deste tipo de sistemas [WWW10]. Neste aspecto, o processo de selecção de vistas multidimensionais assume, de facto, um papel de grande importância.

A partir do controlo das sessões OLAP, muitas vantagens e aplicações se podem desenvolver. A que foi abordada ao longo dos trabalhos de dissertação, foi a possibilidade de corresponder um conjunto de vistas que se identifica por possuir uma forte probabilidade de ser consulta. Tendo em conta as vantagens que se podem tirar a partir de uma selecção de vistas baseada nos padrões de exploração OLAP, planeou-se um método de selecção que constrói um *iceberg cube* composto pelas vistas com maior frequência de consultada ao longo das sessões OLAP. O método

desenvolvido denomina-se de M3. Este método assenta-se essencialmente na construção de estruturas que relacionam a frequência de utilização das vistas, reflectida em termos de coloração e de probabilidades, que foram consultadas durante um determinado conjunto de sessões OLAP, realizadas por um ou mais utilizadores numa plataforma analítica. Estas permitem identificar qual o grupo de vistas (*cuboids*) que integram a *lattice* de um cubo que se evidenciaram mais em termos de processos de consulta e na ordem com que foram consultadas – a sua sequência de consulta. Se o primeiro destes aspectos se resume a uma simples contagem de *hits* feitos pelas *queries* sobre os *cuboids*, que é traduzida posteriormente para uma representação colorida, o segundo aspecto é extraído de uma Cadeia de Markov especificamente desenvolvida para a caracterização de perfis de utilização OLAP. Em conjugação, estes dois aspectos de análise de frequência e sequência de consultas, mais as restrições prévias de materialização obrigatórias, permitem-nos estabelecer o mais adequado conjunto de vistas a materializar para um dado conjunto de utilizadores, para um determinado período de utilização.

De referir, que o método M3 revelou-se bastante efectivo nos testes laboratoriais desenvolvidos para a validação da sua execução e para prova da sua viabilidade. Mas para executar uma avaliação concreta do método de selecção M3, foram definidos dois esquemas dimensionais, que visam perceber o seu desempenho e perceber a sua performance de forma mais precisa. Muito embora o método M3 possua um pressuposto de selecção que facilmente se identifica como sendo vantajoso, identificou-se a necessidade de observar, em valores concretos, o quão vantajoso se torna esta abordagem [WWW06]. Para além de avaliar o método M3, comparou-se também os seus desempenhos com outros algoritmos de selecção (PBS [Shuklaet al. 1998] e o HRU [Hanusseet al. 2009]). Sobre cada método de selecção foram testados e registados três aspectos fulcrais que determinam a qualidade de um método de selecção, que são: os seus tempos de execução para indicarem um conjunto de vistas para ser materializado; a memória efectivamente gasta tendo em conta o limite de espaço máximo, que é passado como parâmetro em cada um dos métodos; e por fim, perceber a qualidade do conjunto de vistas indicado por cada um dos métodos. Este último teste consistiu em efectuar a simulação de trezentas *queries* de consulta, baseadas nos conteúdos anteriormente consultados e contabilizar o número de vezes em que o servidor OLAP possuía materializado os valores pretendidos nas consultas, assim como o número de vezes que tal não se verificou (em que houve a necessidade de consultar o *data warehouse* para responder à consulta). O método M3 demonstrou-se muito vantajoso nos dois casos em que foi executado. Ficou comprovado, também com recurso aos testes aqui demonstrados, mas também pelo entendimento que facilmente se pode concretizar, que os pressupostos de selecção seguidos pelo método M3 são de facto muito vantajosos.

A principal conclusão que se pode retirar com os trabalhos realizados, é a de que a materialização de vistas seleccionadas a partir dos padrões de exploração OLAP, permite aumentar consideravelmente a performance das ferramentas analíticas de dados, porque os dados que mais frequentemente são consultados encontram-se materializados no servidor OLAP.

5.2 Trabalhos Futuros

Como vimos, a monitorização de sessões OLAP pode ser uma fonte de informação muito importante na identificação dos conteúdos de um cubo de dados mais relevantes para uma determinada comunidade de utilizadores. Com o conhecimento e a experiência adquirida nesta dissertação, podemos enunciar alguns novos.

Trabalhos que poderão ser realizados no seguimento deste que acabámos de concluir. Salientemos então o seguinte:

- **Identificação de conteúdos desnecessários** - Da mesma forma que a selecção de vistas multidimensionais, que correspondem directamente aos conteúdos mais consultados, trazem vantagens na sua materialização no servidor OLAP, uma vez que melhoram o seu desempenho, de forma similar, a identificação de conteúdos que nunca são consultados revela-se de grande utilidade. Desta forma, pode-se criar uma base que identifique conteúdos que se revelam inúteis e sem necessidade de se manterem, e a partir dela definir processos especialmente concebidos para a sua "eliminação", reduzindo assim o tempo de processamento e memória dedicada à sua materialização e manutenção no servidor OLAP. Ao se remover de um esquema dimensional atributos de uma hierarquia (ou hierarquias inteiras), atributos de dimensões (ou até mesmo dimensões de forma integral), ou medidas de uma tabela de factos, ter-se-á feito uma enorme poupança no processamento relacionado com a alimentação deste tipo de conteúdos no respectivo *data warehouse*, bem como nos recursos adstritos aos respectivos processos de ETL que asseguram a sua actualização [Nicola et al. 2003]. De certa forma, estes processos já se encontram definidos nesta dissertação. O que falta é, basicamente, efectuar a troca de prioridades a atribuir ao conjunto de vistas: deve então ser realçado o que não objecto de consulta durante as sessões de exploração sobre um cubo de dados, ao contrário do que acontece no M3 (onde são seleccionados os conteúdos que são mais consultados).

- **Selecção de segmentos de vistas multidimensionais** - O método de selecção apresentado nesta dissertação gera um cubo *iceberg* com vistas integrais. Uma característica que certos algoritmos de selecção possuem, é elaborar a indicação de segmentos. Ao invés de indicar a vista completa para materialização no servidor OLAP, é dada a indicação para materializar apenas uma secção da vista. Desta forma, pretende-se definir um método de selecção que indique, para além das vistas multidimensionais mais consultadas, as partes que se revelam mais importantes e frequentemente consultadas. Mais uma vez, certos processos podem perfeitamente ser aproveitados desta dissertação. Falta acrescentar níveis de detalhe sobre os conteúdos consultados, em vez de indicar uma materialização total da vista multidimensional.

- **Identificação de sequências de exploração** - Uma das estruturas implementadas no método M3, é a cadeia de Markov que reflecte as sessões de consulta efectuadas sobre o cubo de dados. Nesta estrutura, encontram-se descriminados as sequências de consulta que cada uma das vistas multidimensionais sofreu ao longo das sessões de exploração. Se forem

identificados sequências exploração com enorme probabilidade de execução, podem ser transportadas para as ferramentas de *front-end* logo no início de sessão. De facto ocorrem situações em que compensa efectuar esta sobrecarga sobre a ferramenta OLAP, constatando melhorias de desempenhos com o uso desta prática, dado que ficam logo disponíveis os dados em que se regista uma maior incidência de consulta. Como referido, a estrutura já se encontra definida, assim como os processos que a criam. Falta encontrar os processos que efectuem a respectiva análise sobre a cadeia de Markov e encontre as sequências de consulta que possuem enorme probabilidade de execução.

- **Definição de perfis de exploração** - Um servidor OLAP que possua em sua posse um conjunto considerável de cubos armazenados, atende vários tipos de pedidos sobre vários tipos de informação. Diversos projectos têm vindo a ser efectuados com o objectivo de encontrar perfis de utilização, para perceber o tipo de informação que uma comunidade de utilizadores procura sobre um cubo de dados. O objectivo principal, é então definir perfis de utilizadores e desta forma compreender as execuções de exploração e obtenção às respostas que se revelam mais importantes. Com a definição da cadeia de Markov, que reflecte as sessões de exploração OLAP sobre um cubo, já se cria uma primeira abordagem sobre o problema. Agora falta estender os processos de forma a poder identificar os perfis de utilização, com base nas cadeias de Markov retiradas a partir das sessões de exploração.

- **Identificação de metodologias de trabalho** - Diversas vezes os problemas possuem resoluções idênticas ou sistemáticas. Caso se verifiquem estes casos, seria de enorme utilidade possuir aplicações que para além de resolver os pedidos às *queries* de consulta, também fosse capaz de identificar metodologias que encaminhassem um determinado utilizador aos valores que de facto possuem importância de serem analisados. Em concreto, obteríamos sistemas de alertas e detenção de anomalias. Alguns dos processos estabelecidos no desenvolvimento do método M3, vão de encontro com a resolução deste problema, como por exemplo na aplicação dos filtros sobre a cadeia de Markov colorida que constroem o cubo *iceberg*.

- **Expressão de preferências** - Um tema sobre o qual se tem efectuado estudos significativos, em que se podem constatar diversas tentativas de solução, é na definição de preferências tendo como base as sessões de exploração OLAP [Golfarelli et al. 2009]. O objectivo é desenvolver indicações sobre os conteúdos que preferencialmente são pesquisados e que estão directamente relacionados com uma *query* executada. Da mesma forma com que se efectua uma monitorização sobre as *queries* submetidas (para obter indicação das vistas mais consultadas), também são necessários controlos similares para identificar os conteúdos formulem uma preferência expressa pelos valores obtidos.

Bibliografia

[Baralis et al. 1997] Elena Baralis, Stefano Paraboschi, and Ernest Teniente. 1997. "Materialized Views Selection in a Multidimensional Database". In Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB '97), Matthias Jarke, Michael J. Carey, Klaus R. Dittrich, Frederick H. Lochovsky, Pericles Loucopoulos, and Manfred A. Jeusfeld (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 156-165.

[Beyer et al. 1999] Kevin Beyer and Raghu Ramakrishnan. 1999. "Bottom-up computation of sparse and Iceberg CUBE". In Proceedings of the 1999 ACM SIGMOD international conference on Management of data (SIGMOD '99). ACM, New York, NY, USA, 359-370.

[Chaudhuri & Dayal 1997] Surajit Chaudhuri and Umeshwar Dayal, "An overview of data warehousing and OLAP technology" ACM SIGMOD Record, vol. 26, March 1997.

[Chaudhuri et al. 1997] Surajit Chaudhuri and Umeshwar Dayal. 1997. "Data warehousing and OLAP for decision support". In Proceedings of the 1997 ACM SIGMOD international conference on Management of data (SIGMOD '97), Joan M. Peckman, Sudha Ram, and Michael Franklin (Eds.). ACM, New York, NY, USA, 507-508.

[Chaudhuri et al. 2001] Surajit Chaudhuri, Umeshwar Dayal, and Venkatesh Ganti. 2001. "Database Technology for Decision Support Systems". Computer 34, 12 (December 2001), 48-55.

[Ching et al. 2006] Wai-Ki Ching and Michael K. Ng. "Markov Chains: Models, Algorithms and Applications (International Series in Operations Research & Management Science)". Springer, 1 edition, December 2005.

[Cios et al. 2007] Krzysztof J. Cios, Witold Pedrycz, Roman W. Swiniarski, and Lukasz A. Kurgan. 2007. "Data Mining: A Knowledge Discovery Approach". Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Codd et al. 1993] E. F. Codd, S. B. Codd, and C. T. Salley, "Providing olap to user-analysts: An it mandate" 1993.

[Coffman et al. 1985] E. G. Coffman, T. T. Kadota, and L. A. Shepp. 1985. "On the Asymptotic Optimality of First-Fit Storage Allocation". IEEE Trans. Softw. Eng. 11, 2 (February 1985), 235-239.

[Connolly & Begg 2001] Thomas Connolly and Carolyn Begg "Database Systems: A Practical Approach to Design, Implementation, and Management (2nd Edition)". Addison Wesley, 2001.

[Cuzzocrea et al. 2009] Alfredo Cuzzocrea and Svetlana Mansmann, "OLAP visualization: models, issues, and techniques", Encyclopedia of Data Warehousing and Mining, 2nd ed., IGI Global, Hershey, PA, USA, pp. 1439-1446, (2009).

[C. Thomsen 2008] C. Thomsen, 2008, "Aspects of Data Warehouse Technologies for Complex Web Data," PhD thesis, Aalborg Universitet.

[E. F. Codd 1970] Edgar Frank Codd. 1970. "A relational model of data for large shared data banks". Commun. ACM 13, 6 (June 1970), 377-387.

[Golfarelli & Rizzi 2009] Matteo Golfarelli and Stefano Rizzi, "Data Warehouse Design: Modern Principles and Methodologies" McGraw-Hill Osborne Media, 2009.

[Golfarelli et al. 2009] Matteo Golfarelli and Stefano Rizzi. 2009. "Expressing OLAP Preferences". In Proceedings of the 21st International Conference on Scientific and Statistical Database Management (SSDBM 2009), Marianne Winslett (Ed.). Springer-Verlag, Berlin, Heidelberg, 83-91.

[Gong et al. 2008] An Gong and Weijing Zhao. 2008. "Clustering-Based Dynamic Materialized View Selection Algorithm". In Proceedings of the 2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery - Volume 05 (FSKD '08), Vol. 5. IEEE Computer Society, Washington, DC, USA, 391-395

[Gunopulos et al. 2000] Dimitrios Gunopulos, George Kollios, Vassilis J. Tsotras, and Carlotta Domeniconi. 2000. "Approximating multi-dimensional aggregate range queries over real attributes". SIGMOD Rec. 29, 2 (May 2000), 463-474.

[Gupta et al. 1997] Himanshu Gupta, Venky Harinarayan, Anand Rajaraman, and Jeffrey D. Ullman. 1997. "Index Selection for OLAP". In Proceedings of the Thirteenth International Conference on Data Engineering (ICDE '97), Alex Gray and Peroke Larson (Eds.). IEEE Computer Society, Washington, DC, USA, 208-219.

[Gupta et al. 1999] Himanshu Gupta and Inderpal Singh Mumick. 1999. "Selection of Views to Materialize Under a Maintenance Cost Constraint". In Proceedings of the 7th International Conference on Database Theory (ICDT '99), Catriel Beeri and Peter Buneman (Eds.). Springer-Verlag, London, UK, 453-470.

[G. Colliat 1996] George Colliat. 1996. "OLAP, relational, and multidimensional database systems". SIGMOD Rec. 25, 3 (September 1996), 64-69

[Hanusse et al. 2009] Nicolas Hanusse, Sofian Maabout, and Radu Tofan. 2009. "A view selection algorithm with performance guarantee". In Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology (EDBT '09), Martin Kersten, Boris Novikov, Jens Teubner, Vladimir Polutin, and Stefan Manegold (Eds.). ACM, New York, NY, USA, 946-957.

[Harinarayan et al. 1996] Venky Harinarayan, Anand Rajaraman, and Jeffrey D. Ullman. 1996. "Implementing data cubes efficiently". In Proceedings of the 1996 ACM SIGMOD international conference on Management of data (SIGMOD '96), Jennifer Widom (Ed.). ACM, New York, NY, USA, 205-216.

[Hasan et al. 2007] K. M. Azharul Hasan, Tatsuo Tsuji, and Ken Higuchi. 2007. "An efficient implementation for MOLAP basic data structure and its evaluation". In Proceedings of the 12th international conference on Database systems for advanced applications (DASFAA'07), Ramamohanarao Kotagiri, P. Radha Krishna, Mukesh Mohania, and Ekawit Nantajeewarawat (Eds.). Springer-Verlag, Berlin, Heidelberg, 288-299.

[H. Gupta 1997] Himanshu Gupta and Inderpal Singh Mumick. 2005. "Selection of Views to Materialize in a Data Warehouse". IEEE Trans. on Knowl. and Data Eng. 17, 1 (January 2005), 24-43.

[I. Molyneaux 2009] Ian Molyneaux. 2009. "The Art of Application Performance Testing: Help for Programmers and Quality Assurance (1st ed.)". O'Reilly Media, Inc..

[Kimball et al. 2008] Ralph Kimball, Margy Ross, Warren Thornthwaite, Joy Mundy, and Bob Becker, "The Data Warehouse Lifecycle Toolkit". Wiley, 2008.

[Kotidis et al. 1999] Yannis Kotidis and Nick Roussopoulos. 1999. "DynaMat: a dynamic view management system for data warehouses". In Proceedings of the 1999 ACM SIGMOD international conference on Management of data (SIGMOD '99). ACM, New York, NY, USA, 371-382.

[Kotsis et al. 2000] Nikolaos Kotsis and Douglas R. McGregor. 2000. "Elimination of Redundant Views in Multidimensional Aggregates". In Proceedings of the Second International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2000), Yahiko Kambayashi, Mukesh K. Mohania, and A. Min Tjoa (Eds.). Springer-Verlag, London, UK, 146-161.

[Koutsoukis et al. 1999] Nikitas-Spiros Koutsoukis, Gautam Mitra, and Cormac Lucas. 1999. "Adapting on-line analytical processing for decision modelling: the interaction of information and decision technologies". Decis. Support Syst. 26, 1 (July 1999), 1-30.

[Lakshmanan et al. 2003] Laks V. S. Lakshmanan, Jian Pei, and Yan Zhao. 2003. "QC-trees: an efficient summary structure for semantic OLAP". In Proceedings of the 2003 ACM SIGMOD international conference on Management of data (SIGMOD '03). ACM, New York, NY, USA, 64-75.

[Lawrence et al. 2006] Michael Lawrence, Andrew Rau-Chaplin. "Dynamic view selection for OLAP". In: Tjoa AM, Trujillo J, eds. Proc. of the 8th Int'l Conf. on Data Warehousing and Knowledge Discovery (DaWaK 2006). Krakow: Springer-Verlag, 2006. 33-44.

[Lijuan et al. 2009] Zhou Lijuan, Ge Xuebin, Wang Linshuang, and Shi Qian. 2009. "Research on Materialized View Selection Algorithm in Data Warehouse". In Proceedings of the 2009 International Forum on Computer Science-Technology and Applications - Volume 02 (IFCSTA '09), Vol. 2. IEEE Computer Society, Washington, DC, USA, 326-329.

[Lin et al. 2007] Ziyu Lin, Dongqing Yang, Guojie Song, and Tengjiao Wang. 2007. "User-Oriented Materialized View Selection". In Proceedings of the 7th IEEE International Conference on Computer and Information Technology (CIT '07). IEEE Computer Society, Washington, DC, USA, 133-138.

[Morfonios et al. 2007] Konstantinos Morfonios, Stratis Konakas, Yannis Ioannidis, and Nikolaos Kotsis. 2007. "ROLAP implementations of the data cube". ACM Comput. Surv. 39, 4, Article 12 (November 2007).

[Nadeau et al. 2002] Thomas P. Nadeau and Toby J. Teorey. 2002. "Achieving scalability in OLAP materialized view selection". In Proceedings of the 5th ACM international workshop on Data Warehousing and OLAP (DOLAP '02). ACM, New York, NY, USA, 28-34.

[Nicola et al. 2003] Nicola and Haider Rizvi. 2003. "Storage Layout and I/O Performance in Data Warehouses. Database". p.1-9.

[Qiu et al. 2000] Shi Guang Qiu and Tok Wang Ling. 2000. "View Selection in OLAP Environment". In Proceedings of the 11th International Conference on Database and Expert Systems Applications (DEXA '00), Mohamed T. Ibrahim, Josef King, and Norman Revell (Eds.). Springer-Verlag, London, UK, 447-456.

[R. Kimball 1997] Ralph Kimball. 1997. "A dimensional modeling manifesto". DBMS 10, 9 (August 1997), 58-70.

[R. Sarukkai 2000] Ramesh R. Sarukkai. 2000. "Link prediction and path analysis using Markov chains". In Proceedings of the 9th international World Wide Web conference on Computer networks: the international journal of computer and telecommunications netowrking. North-Holland Publishing Co., Amsterdam, The Netherlands, The Netherlands, 377-386.

[Sapia et al. 1998] Carsten Sapia, Markus Blaschka, Gabriele Hofling, and Barbara Dinter. 1998. "Extending the E/R Model for the Multidimensional Paradigm. In Proceedings of the Workshops on Data Warehousing and Data Mining: Advances in Database Technologies" (ER '98), Yahiko Kambayashi, Dik Lun Lee, Ee-Peng Lim, Mukesh K. Mohania, and Yoshifumi Masunaga (Eds.). Springer-Verlag, London, UK, 105-116

[Shukla et al. 1998] Amit Shukla, Prasad Deshpande, and Jeffrey F. Naughton. 1998. "Materialized View Selection for Multidimensional Datasets". In Proceedings of the 24rd International Conference on Very Large Data Bases (VLDB '98), Ashish Gupta, Oded Shmueli, and Jennifer Widom (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 488-499.

[Sismanis et al. 2002] Yannis Sismanis, Antonios Deligiannakis, Nick Roussopoulos, and Yannis Kotidis. 2002. "Dwarf: shrinking the PetaCube". In Proceedings of the 2002 ACM SIGMOD international conference on Management of data (SIGMOD '02). ACM, New York, NY, USA, 464-475.

[T. Palpanas 2000] Themistoklis Palpanas. 2000. "Knowledge discovery in data warehouses". SIGMOD Rec. 29, 3 (September 2000), 88-100

[Valluri et al. 2002] Satyanarayana R Valluri, Soujanya Vadapalli, and Kamalakar Karlapalem. 2002. "View relevance driven materialized view selection in data warehousing environment". Aust. Comput. Sci. Commun. 24, 2 (January 2002), 187-196.

[Vitter et al. 1999] Jeffrey Scott Vitter and Min Wang. 1999. "Approximate computation of multidimensional aggregates of sparse data using wavelets". In Proceedings of the 1999 ACM SIGMOD international conference on Management of data (SIGMOD '99). ACM, New York, NY, USA, 193-204.

Referências WWW

[WWW01] http://www.dwinfocenter.org/

Em vigor desde 1995, este *website* tem como objectivo principal, tem como principal objectivo, o de esclarecer quais os pontos fortes e fracos que verificados em sistemas de *data warehousing*. Acedido em 11 de Julho de 2011.

[WWW02] http://www.tech-faq.com/data-warehouse.html

Website que tem como objectivo efectuar esclarecimentos sobre as questões que mais frequentemente surgem em torno das utilidades que advêm da implementação de um *data warehouse*. Acedido em 17 de Julho de 2011.

[WWW03] http://www.olaphouse.com/documents/olap/why-olap.xml?lang=en

Página oficial da empresa OLAPHouse, que efectua implementações de sistemas OLAP e que promove a sua utilização. Acedido em 17 de Julho de 2011.

[WWW04] http://www.daniel-lemire.com/OLAP/index.html

Website onde se encontram diversas publicações de artigos, fortemente relacionados com sistemas de *data warehousing* e aplicações OLAP. Acedido em 20 de Julho de 2011.

[WWW05] http://infolab.stanford.edu/warehousing/warehouse.html

O projecto *WareHouse Information Prototype at Stanford* (WHIPS) teve como objectivo primordial analisar a criação e a manutenção de sistemas de *data warehousing* e desenvolver algoritmos e ferramentas que assegurem estas actividades. Aqui encontra-se toda a documentação disponível sobre este projecto. Acedido em 20 de Julho de 2011.

[WWW06] http://msdn.microsoft.com/en-us/library/bb924375.aspx

Website mantido directamente pela Microsoft, em que se apresentam planos e são abordados procedimentos que visam efectuar testes de performance, vocacionado fortemente para aplicações *web*. Acedido em 21 de Setembro de 2011

[WWW07] http://www.perftestplus.com/pubs.htm

Website onde se encontram diversas publicações de artigos e trabalhos relacionados com o desenvolvimento de testes de performance em aplicações de software. Acedido em 21 de Julho de 2011.

[WWW08] http://www.oracle.com/technetwork/database/options/olap/index.html

Website mantido directamente pela Oracle, onde se efectua uma explicação muito breve do que são as ferramentas analíticas de dados e as suas implicações. Acedido em 5 de Setembro de 2011.

[WWW09] http://www-01.ibm.com/software/data/cognos/olap-reporting.html

Website mantido directamente pela IBM, onde se efectua o enquadramento das ferramentas OLAP como ferramenta de *reporting* e de análise de dados. Acedido em 5 de Setembro de 2011.

[WWW10] http://www.bi-verdict.com/

Website onde se encontram efectuadas análises sobre aplicações da área de *business intelligence*, realizadas pela equipa de consultores BARC (Business Application Research Center) que se dedica na compra, no desenvolvimento de testes de qualidade e no fornecimento de informação sobre produtos relacionados com *business intelligence.*