Avaliação de Técnicas para Redução de Base de Dados de Produção

Edward Alves R. Neto[1], André Assis Lôbo de Oliveira[1], Plínio de Sá L. Junior[1], Celso G. Camilo Junior[1], Cassio Leonardo Rodrigues[1], Auri Marcelo R. Vincenzi[1]

[1]Instituto de Informática – Universidade Federal de Goiás (UFG)

Caixa Postal 131 – 74.001-970 – Goiânia – GO – Brasil.

{edward_neto,andreoliveira,plinio,celso,cassio,auri}@inf.ufg.br

Abstract. *This paper focuses on the evaluation of techniques for reducing Database Production using SQL queries to software testing. In the context was selected two approaches, the first uses reduction for queries coverage and the second uses Genetic Algorithms. Was used a Benchmark as common database and SQL Mutation to evaluate the test data generated by the techniques. The experiments compares results in terms of efficiency (runtime) and effectiveness (quality of data). The results show that data quality is maintained proportionally by the size of the base test.*

Resumo. *Este artigo centra-se na avaliação de técnicas para redução de Bancos de Dados de Produção, utilizando instruções SQL para o teste de software. No contexto foram utilizadas duas abordagens, a primeira utiliza redução por cobertura de instrução e a segunda faz o uso de Algoritmos Genéticos. Utilizou-se um Benchmark como base de dados comum e a Análise de Mutantes SQL para avaliar os dados de testes gerados pelas técnicas. A experimentação compara os resultados em termos de eficiência (tempo de execução) e eficácia (qualidade dos dados). Os resultados revelam que a qualidade dos dados é mantida proporcionalmente pelo tamanho da Base de Dados de Teste.*

1. Introdução

A obtenção da qualidade de software tem sido o alvo de muitas empresas e pesquisas do segmento de desenvolvimento de software. Por isso, atividades de Validação, Verificação e Teste (VV&T) devem ser executadas durante todo processo de construção de um software [Delamaro, Jino e Maldonado, 2007]. Dentre as atividades de VV&T, o Teste de Software tem grande importância por fazer uso de técnicas e critérios de teste para avaliação da adequabilidade de um conjunto de teste. Dentre os diferentes critérios existentes destaca-se o Teste de Mutação (TM, ou Análise de Mutantes), por ser reconhecidamente eficaz na revelação de defeitos.

Assim como qualquer código, instruções de consulta SQL (Structured Query Language) precisam ser testadas para garantir a qualidade do produto a ser entregue. A Análise de Mutantes SQL (AMS) é uma abordagem do TM para testar as instruções SQL que serão utilizadas na base de dados.

Todavia, o alto custo computacional advindo da execução dos mutantes é um dos grandes problemas de aplicabilidade do TM devido a grande cardinalidade do conjunto de teste. Por isso, manter um banco de dados de testes (BDT) tão pequeno quanto possível tem grande importância [Tuya, 2009], uma vez que um conjunto de teste reduzido diminui o custo da aplicabilidade do critério. Por outro lado, espera-se também que um BDT de tamanho reduzido tenha um conteúdo adequado [Queiroz, 2013] para não perder a qualidade dos testes.

Neste contexto, pesquisas vêm sendo realizadas para propor técnicas que obtenham eficácia e eficiência. O objetivo desta pesquisa consiste em avaliar diferentes técnicas de redução de Banco de Dados em termos de eficácia (qualidade) e eficiência (tempo gasto para redução).

Na avaliação foram consideradas duas abordagens de redução: i) Algoritmos Genéticos (AG) [Monção, 2013] e; ii) a redução por Consulta Consciente (Query-Aware) [Tuya, 2009]. Além disso, para avaliar a qualidade das reduções, foi utilizado um Benchmark desenvolvido especificamente para avaliação de técnicas de redução de Banco de Dados.

Este artigo está estruturado da seguinte forma: a Seção 2 apresenta uma revisão teórica da Análise de Mutantes e expõe as vantagens da redução de bancos de dados para fins de testes; a Seção 3 apresenta as abordagens que serão utilizadas, bem como o motivo de seleção das mesmas; a Seção 4 apresenta os resultados obtidos; a Seção 5 realiza as conclusões e apresenta os trabalhos futuros.

2. Revisão

2.1 Análise de Mutantes

A Análise de Mutantes ou Teste de Mutação é um critério de teste da técnica baseado em defeitos. Surgiu em 1978 na Universidade de Yale com o artigo *"Hints on test data selection. Help for the practicing programmer"* que apresentaram duas principais ideias: a hipótese do programador competente e o efeito de acoplamento. A hipótese do programador competente adota que bons programadores escrevem códigos muito próximos de estarem corretos. Acerca da afirmação, então, assume-se que os erros acontecem por pequenos desvios sintáticos que alteram a semântica do programa. E o efeito de acoplamento diz que defeitos complexos são causados por pequenos erros.

No Teste de Mutação, dado um programa P, são gerados n programas P' (mutantes) que foram modificados sintaticamente utilizando operadores de mutação. O caso de teste é executado no programa original e no mutante visando o resultados gerados. Se a saída do mutante for diferente do programa original, diz-se então que o mutante foi morto. Há casos em que são gerados mutantes equivalentes que, embora sintaticamente sejam diferentes, são semanticamente equivalentes ao programa original.

Após a execução dos casos de testes em todos os mutantes, realiza-se o cálculo do escore de mutação (EM). O valor do escore varia de 0 a 1 e é calculado através de uma relação entre os mutantes gerados e os mutantes mortos. Ele fornece uma medida objetiva de quanto o conjunto de casos de teste analisado aproxima-se da adequação [Delamaro, Jino e Maldonado, 2007], representado pela equação:

$$ms(P,T) = \frac{DM(P,T)}{M(P) - EM(P)}$$

Onde:
ms(P, T): escore de mutação do conjunto de testes T;
DM(P,T) : número de mutantes mortos pelo conjunto de casos de teste T;
M(P): número total de mutantes gerados a partir do programa P; e
EM(P) número de mutantes equivalentes a P;

O principal problema no TM consiste na possibilidade de geração de um grande número de mutantes, o que irá demandar de um alto esforço e custo computacional para que os mutantes sejam executados contra todos os casos de teste.

A Análise de Mutantes SQL é uma técnica que utiliza o TM como abordagem, sendo análoga ao Teste de Mutação tradicional, os programas nesse caso são as instruções SQL e o conjunto de casos de teste T pode ser considerada como todo o Banco de Dados de Produção (BDP). O objetivo também é análogo ao TM, consistindo em formar uma BDT reduzida com alto escore de mutação.

2.2 Redução de Banco de Dados (BD)

No contexto de testes de instruções SQL, o domínio de entrada para a execução dos testes pode possuir mais de um banco de dados e o uso do BDP para testes pode ter um alto custo computacional considerando o tamanho do BDP e complexidade das instruções SQL [Monção, 2013]. A redução de uma base de dados baseia-se na seleção de um número menor de tuplas que possa representar o BDP de forma mais adequada possível.

Diversas técnicas vêm sendo aplicadas no problema de redução de Banco de Dados para o teste de instruções SQL. Manilla e Raiha (2006) propõem a geração automática de um BDT de determinada instrução SQL. Gupta et al (2010) trabalham com a geração de BDT para matar mutantes SQL considerando as mutações nas cláusulas JOIN e nos operadores relacionais. Monção (2013) propôs a geração de BDTs utilizando computação evolucionária com o uso de AG, evoluindo os dados de testes até serem os mais adequados possíveis. Tuya et al. (2009) desenvolveram uma ferramenta para redução de banco de dados através da cobertura de instruções SQL.

Tuya et al. propuseram uma técnica que mantém o foco na extração de uma quantidade mínima de tuplas, preservando a cobertura das instruções SQL, baseada nos critérios de cobertura MC/DC [Tuya, 2010]. Foi desenvolvida uma ferramenta para automatizar o processo denominada QAShrink (Query-Aware Shrinking, do inglês Redução por Consulta Consciente).

Um estudo de caso foi apresentado e obteve como resultado, a redução de um BDP com 137.490 tuplas em 31 tabelas para apenas 223 linhas, uma redução de 99,84%. O resultado também mostrou que o escore de mutação do BDT foi de apenas 0,5% menor em relação ao original e sendo executado no tempo total de 125,2 segundos. Percebe-se que a relação eficiência/eficácia do método é bastante promissora.

Monção (2006) trabalhou com Algoritmos Genéticos, uma meta-heurística a fim de explorar a busca selecionando melhores indivíduos candidatos para compor o BDT. Para a avaliação dos dados selecionados utilizou-se da Análise de Mutantes para SQL. Foram desenvolvidos os AG Canônico (AGCA), AG com Grupo de Eleitos (AGGE) e o InVitro (AG InVitro). A estrutura do AG pode ser representada pela figura 1:

Figura 1. Estrutura de execução do AG.

```
início
   t ← 0
   inicialize P(t)
   avalie P(t)
   enquanto (não condição de parada) faça
   início
      t ← t + 1
      selecione P(t) a partir de P(t – 1)
      altere P(t)
      avalie P(t)
   fim
fim
```

3. Abordagem Proposta

A presente pesquisa tem por objetivo avaliar diferentes técnicas de redução de Banco de Dados. Todavia, para alcança-la, existe a necessidade de um Benchmark que permita a avaliação das técnicas selecionadas.

Queiroz (2013) propôs um Benchmark[1] para o auxílio na avaliação de diferentes técnicas de redução de BDP. O Benchmark possui dois cenários, o primeiro utiliza um BD (Banco de Dados) hipotético e o segundo utiliza a aplicação de um BD real, além disso possui um BDE (Banco de Dados de Experimentos) que contém as informações obtidas a partir da redução pelo método aleatório de busca. A escolha de um Benchmark auxilia na avaliação por comparação de diferentes métodos, uma vez que, dispõe de um ambiente comum para o uso das técnicas. Os experimentos deste artigo utilizam o cenário 2 (tabela 1) como base comum para os experimentos de testes das ferramentas.

Tabela 1. Informações do cenário 2 do Benchmark

BDP UFG	
Tabela	Tuplas
histórico	232485
disciplinas	23106

Foram selecionadas 4 consultas (instruções 1, 4, 8 e 15) baseadas no espaço de melhoria médio, que é a diferença entre o escore de mutação dos BDTs do Método Aleatório e o escore de mutação do BDP do Benchmark, e instruções que utilizassem somente uma tabela na consulta, pois o AG selecionado não oferece suporte para instruções com mais de duas tabelas na consulta. As instruções são:

1) select codTurma, nomecampus from histórico where nomeCurso like 'Mestrado em Química%';
4) select chave from historico where professor = 1597;
8) select * from historico where professor = 5962 and nomeCurso = 'Zootecnia' and situacao = 'aprovado'; e
15) select discente, situacao from historico where disciplina = 22 and nomeCampus <> 'Campus de Catalão' and situacao = 'Aprovado' and discente not in (78661, 80836, 79782, 86615, 85339).

A pesquisa utiliza as técnicas de redução por Consulta-Consciente (QAShrink) [Tuya, 2009] e redução por Algoritmos Genéticos [Monção, 2013], para serem avaliadas em termos de escore de mutação (eficácia) e tempo de execução (eficiência).

Na primeira abordagem, somente são necessários como parâmetros de entrada as instruções SQL e o Banco de Dados de Produção que deseja ser reduzido, além disso foi necessária a criação de um BDT somente com a estrutura do BDP, para que pudessem ser armazenados os dados após a redução. As instruções selecionadas utilizam somente a tabela "histórico" do cenário 2 e foram executadas isoladamente umas das outras.

A execução da primeira abordagem gerou BDTs de tamanhos diferentes para cada instrução selecionada. Para a instrução 1 e 4, o método reduziu o BDP para apenas 2 tuplas, já com a instrução 8 a redução foi para 4 tuplas e a instrução 15 com 5 tuplas. Em média as reduções foram de cerca de 99% para todos os casos.

[1] Considera-se um *Benchmark* como sendo a composição de uma estrutura (cenário) juntamente com um conjunto de instâncias de problemas [Queiroz, 2013].

A partir da execução da primeira técnica, proveu-se dos dados de entrada para a realização dos testes com o AG. Para que os métodos pudessem ser comparados, o

tamanho do BDT gerado pelo QAShrink foi o parâmetro de entrada para a realização dos testes com o AG. Nesse contexto o tamanho do cromossomo (que é o tamanho do BDT que se deseja) foi ajustado para que recebesse o valor do tamanho do BDT gerado pela primeira abordagem. A tabela 2 apresenta os valores utilizados:

Tabela 2. Parâmetros para execução do AG

Execução do AGCA			
Parâmetros Fixos		Parâmetros Variáveis	
Tx. de cruzamento (%)	80	Instrução	1, 4, 8, 15
Tam. população	45		
Qtde. gerações	50	Tam. do cromossomo (tamanho do BDT)	2 (instrução 1 e 4)
Op. cruzamento	1		4 (instrução 8)
Elitismo	2		5 (instrução 15)
Escore do BDP	1	Tx. de Mutação (%)	5, 10, 20, 30 e 40
Qtde. experimentos	10		

A função objetivo do AG para cálculo do *fitness* (ou escore de mutação) foi adaptada para realizar a avaliação dos BDTs provenientes do QAShrink.

4. Resultados

Os resultados das execuções com o QAShrink e do AGCA utilizando o cenário 2 do benchmark estão expressos nos gráficos, 1, 2, 3 e 4. Todos os resultados consistem na média de 10 execuções dos métodos. A primeira fase dos resultados consiste na análise dos escores de mutação (eficácia) de cada técnica em cada problema.

Gráfico 1. Resultados da instrução 1

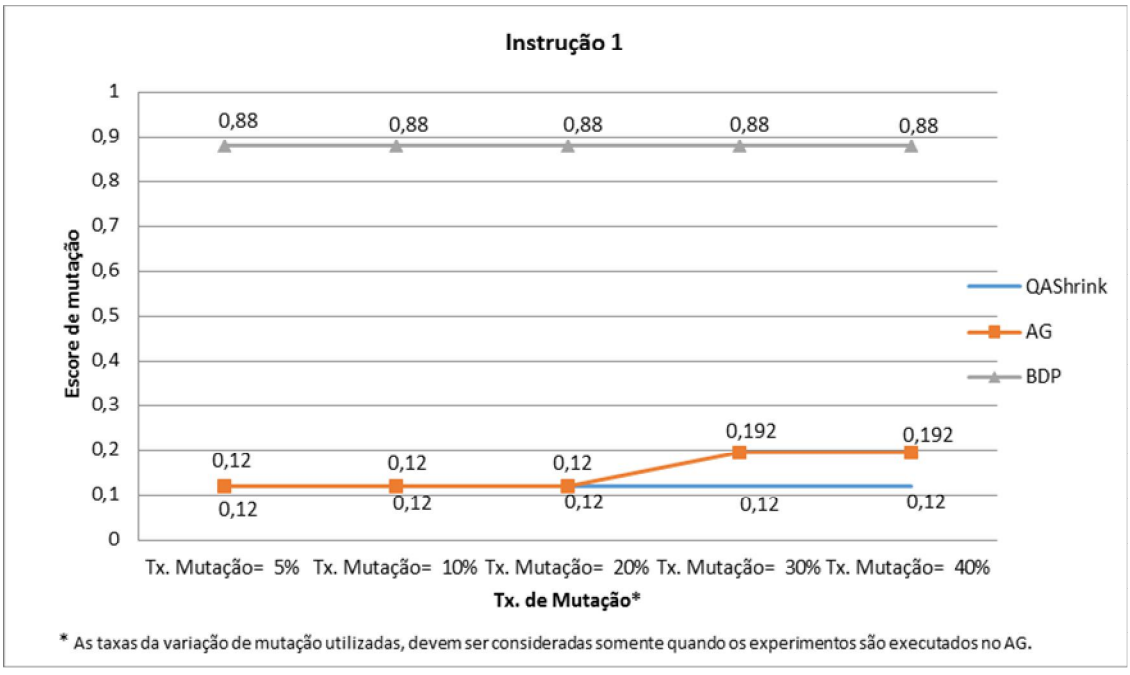


Gráfico 2. Resultados da instrução 4

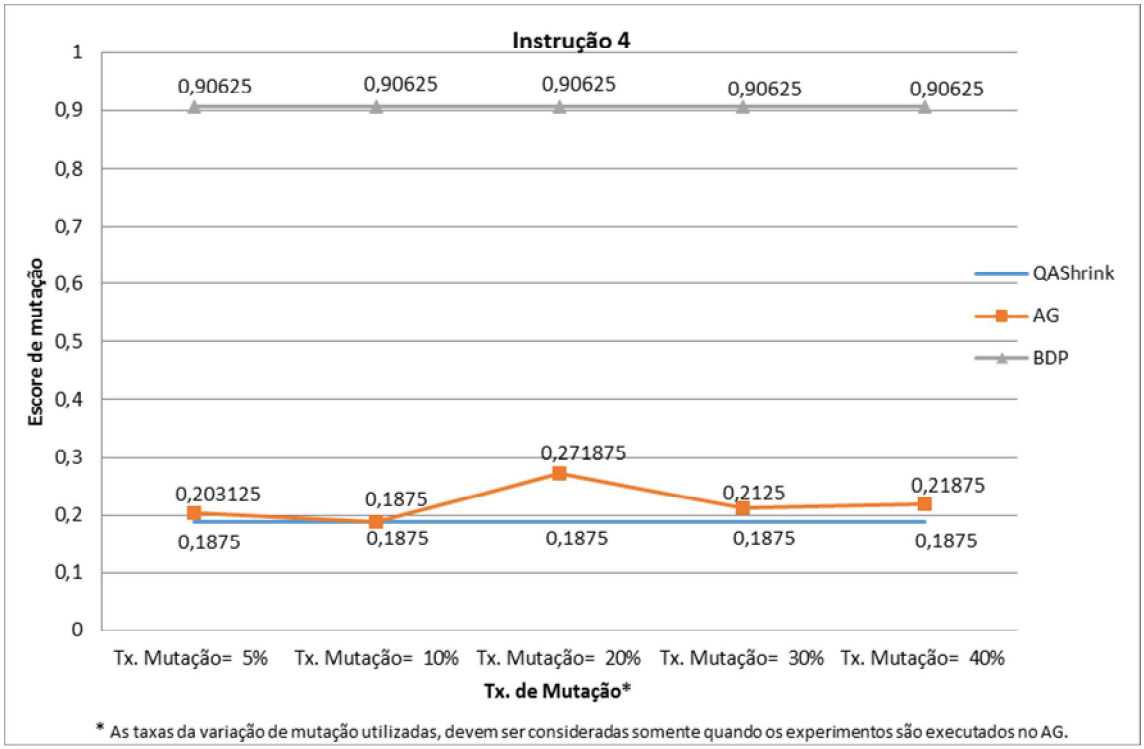


Gráfico 3. Resultados da instrução 8

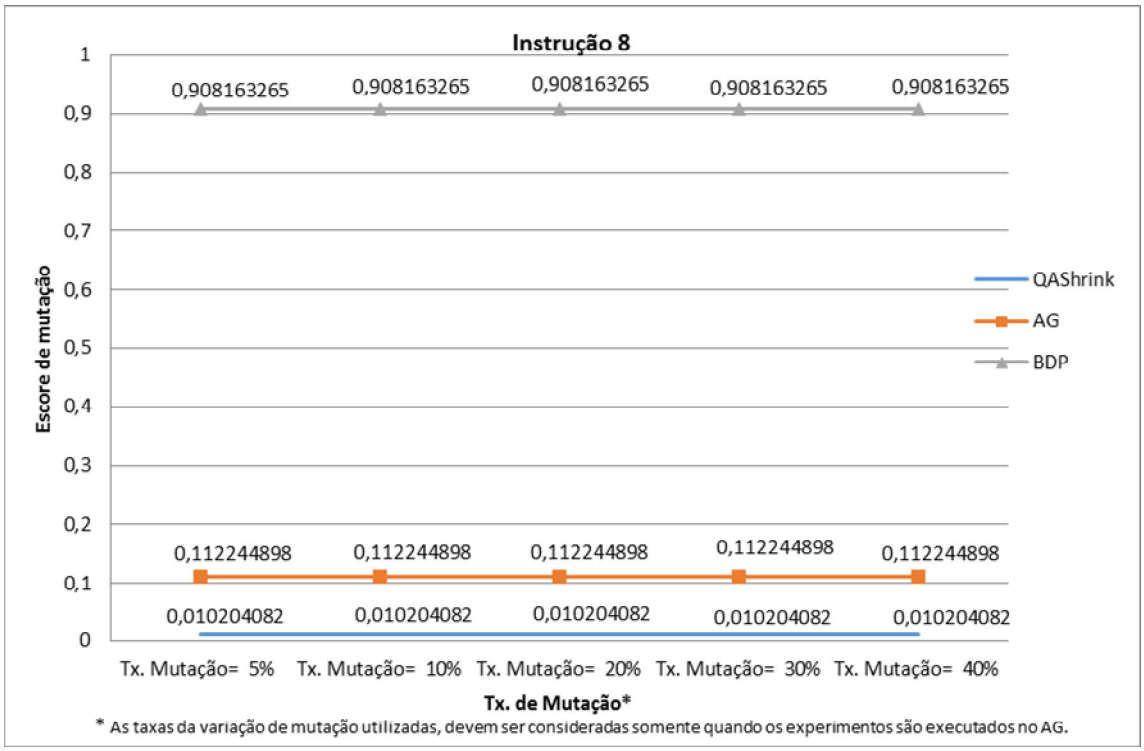
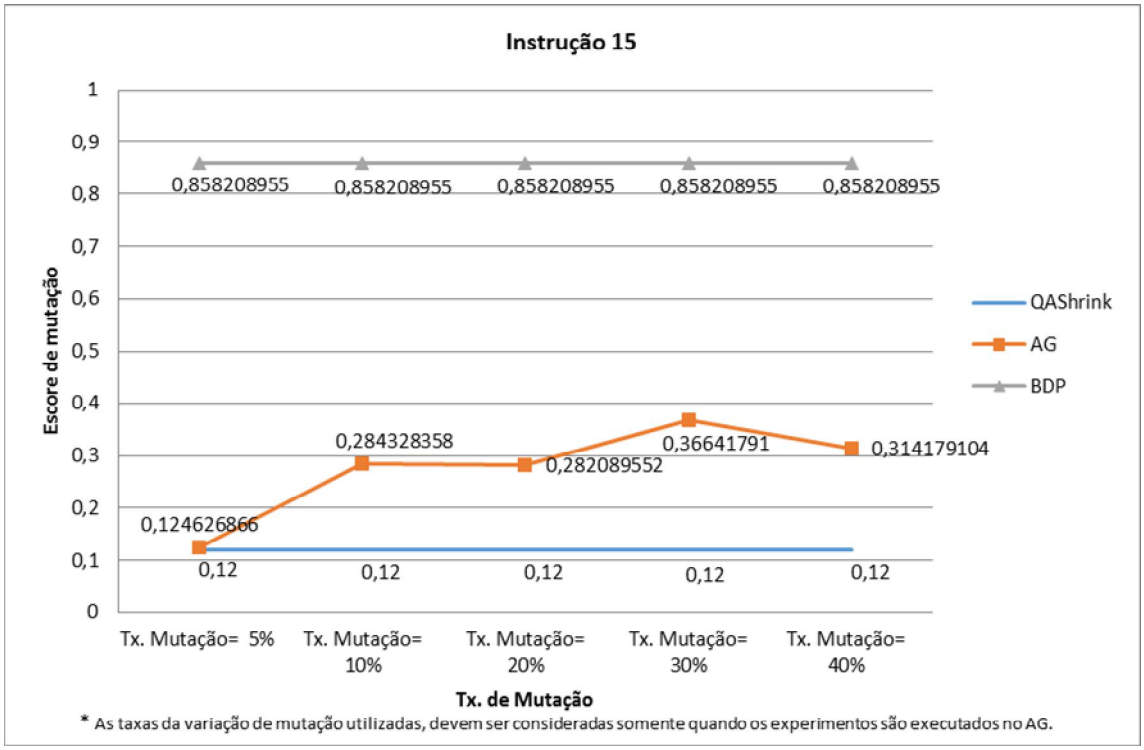


Gráfico 4. Resultados da instrução 15



Em geral, os dados de teste gerados pelo AG obtiveram um melhor escore de mutação na maioria dos casos. A instrução 15 foi a que obteve maior diferença de escores entre as duas técnicas. Nesses casos percebe-se que a taxa de mutação é o parâmetro que causa essa diferença, quanto maior a taxa maior a diferença entre os escores.

Bons escores de mutação são aqueles que têm o valor elevado, mais próximo do BDP possível. Considerando as instruções utilizadas nota-se que o valor dos EM são bastante baixos. Uma possível explicação para isso consiste no fato de que foi realizada no primeiro método uma redução de mais de 99% em todos os casos, o que interferiu na qualidade dos escores obtidos, assim as instâncias alcançadas não podem ser consideradas representativas em relação ao BDP original.

A tabela 3 apresenta o tempo médio gasto na execução de cada experimento.

Tabela 3. Tempos de execução das ferramentas (em segundos)

Instrução	QAShrink	AG (variação da tx. de mutação)				
		5%	10%	20%	30%	40%
1	3,33	1510,04	1045,11	1000,67	1194,21	1065,15
4	2,22	1156,21	1639,83	1347,02	1438,49	1564,11
8	2,69	1536,83	1683,93	1679,61	1953,03	1953,06
15	2,18	2340,20	4260,47	3252,35	3677,34	3347,82

Analisando a tabela 3 obtemos que, em termos de tempo de execução a técnica QAShrink realizou a busca por BDTs em um tempo bastante inferior que o AG. Todavia, percebe-se que o AG na maioria dos casos obteve maiores escores de mutação, ou seja, a primeira abordagem obteve uma eficiência (tempo para execução) melhor enquanto a segunda foi melhor em termos de eficácia (qualidade dos dados de teste).

5. Conclusões e Trabalhos Futuros

Neste artigo foram avaliadas duas técnicas de redução de Banco de Dados de Produção: redução por Consulta-Consciente [Tuya, 2009] e redução por Algoritmos Genéticos [Monção, 2009]. Para avaliação dos dados de teste gerados foi realizada a Análise de Mutantes SQL em um Benchmark específico para a avaliação de técnicas de redução de Banco de Dados [Queiroz, 2013].

As técnicas foram avaliadas sob a perspectiva de Escore de Mutação (Análise de Mutantes SQL) para avaliar a qualidade das reduções (eficácia), bem como sobre o tempo gasto para obter a redução (eficiência). Na primeira técnica, obteve-se um tempo de execução bastante inferior em relação a segunda, já o AG obteve uma melhor qualidade dos dados obtidos. Com base nos resultados, podemos concluir que uma grande redução no BDP pode prejudicar a qualidade dos dados de teste.

Como trabalhos futuros pretende-se propor uma abordagem que seja capaz de identificar o tamanho ideal para um banco de dados de testes (BDT), a fim de que tal redução não comprometa a sua qualidade para revelação de defeitos.

Referências

Delamaro, M. E.; Jino, M.; Maldonado, J. C. Introdução ao Teste de Software. Campus, 2007.

DeMillo, R. J Lipton, and F. G. Sayward. “Hints on test data Selection: help for the practicing programmer”. In IEEE Computer Society Press Los Alamitos, p. 34-41, 1978.

Gupta, B. P.; Vira, D; Sudarshan, S. X-data: Generating test data for killing sql mutants. In: Li, F. et al. (Ed.). Proceedings of the 26th International Conference on Data Engineering, ICDE 2010.

Manilla, H; Raiha, K. J. Test Data for Relational Queries. In: Proceedings of the fifth ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, 1986.

Monção, A. C. B. L.; Camilo Junior, C. G.; Queiroz, L. T.; Rodrigues, C. L.; Leitao-Junior, P. S.; Vincenzi, A. M. R. “Shrinking a Database to Perform SQL Mutation Tests Using na Evolutionary Algorithm”. In IEEE Congresso n Evolutionary Computation, 2013.

Monção, A. C. B. L. “Uma Abordagem Evolucionária para o Teste de Instruções SELECT SQL com o uso da Análise de Mutantes”, 2013.

Queiroz, L. T. “Um Benchmark para Avaliação de Técnicas de Busca no Contexto de Análise de Mutantes SQL” 2013.

Tuya, Javier; Suárez-Cabal, José M.; Riva, Claudio de la. “Query-Aware Shrinking Databases”. In: 2nd International Workshop on Testing Databases Systems, 2009.

Tuya, Javier; Suárez-Cabal, José M.; Riva, Claudio de la. “Full predicate coverage for testing SQL database queries”. In: Software Testing, Verification and Reliability, 2010.