RAFAEL ANTONIO COVRE

WEDRING: *PIPELINE* PARA ANÁLISE DE EXPRESSÃO DIFERENCIAL EM EXPERIMENTOS DE RNA-SEQ

Dissertação apresentada como requisito parcial para a obtenção do título de Mestre em Bioinformática pelo Programa de Pós-Graduação em Bioinformática, Setor de Educação Profissional e Tecnológica, Universidade Federal do Paraná, área de concentração Bioinformática.

Orientadora: Professora Doutora Rose Adele Monteiro

Coorientador: Professor Doutor Adriano Barbosa da Silva

CURITIBA

2013

Covre, Rafael Antonio
C873 Wedring: pipeline para análise de expressão diferencial em experimentos de RNA-Seq / Rafael Antonio Covre. - Curitiba, 2013.
87 f.: il., tabs, grafs.

Orientadora: Profª Drª Rose Adele Monteiro
Coorientador: Prof. Dr. Adriano Barbosa da Silva
Dissertação (Mestrado) – Universidade Federal do Paraná, Setor de Educação Profissional e Tecnológica, Curso de Pós-Graduação em Bioinformática.

1. Expressão gênica. 2. Nitrogênio - Fixação. 3. Azospirillum brasilense fp2. 4. Genoma. 5. Bioinformática. I. Monteiro, Rose Adele. II. Silva, Adriano Barbosa da. III.Título. IV. Universidade Federal do Paraná.

CDD 575.113

TERMO DE APROVAÇÃO

RAFAEL ANTONIO COVRE

Wedring: Pipeline para analise de expressão diferencial em experimentos de RNA-SEQ

Dissertação aprovada como requisito parcial para obtenção do grau de Mestre em Bioinformática, pelo Programa de Pós-graduação em Bioinformática, Setor de Educação Profissional e Tecnológica, da Universidade Federal do Paraná, pela seguinte banca examinadora:

Orientadora: Profª Drª Rose Adele Monteiro

Profª Drª Michelle Zibetti Tadra Sfeir
Universidade Federal do Paraná - UFPR

Prof. Dr.Helisson Faoro
Universidade Federal do Paraná - UFPR

Curitiba, 14 de maio de 2013

À Bárbara, minha amada

Aos meus pais

Aos meus sogros

E a todos os meus amigos e amigas

que me apoiaram nesses dois anos

AGRADECIMENTOS

Agradeço primeiramente aos meus orientadores, Professora Doutora Rose Adele Monteiro e Professor Doutor Adriano Barbosa da Silva, por terem aceitado a orientação do presente trabalho, tal como por terem se disponibilizado a oferecer tempo e experiência profissional sem a qual seria impossível a realização deste.

Agradeço ao Núcleo de Fixação de Nitrogênio do Departamento de Bioquímica e Biologia Molecular da Universidade Federal do Paraná pela disponibilização dos dados do transcriptoma de *Azospirillum brasilense* fp2.

Agradeço ao Programa de Pós-Graduação em Bioinformática da Universidade Federal do Paraná pela oportunidade oferecida, tal como pela concessão do acesso ao *cluster* da Bioinformática, o qual foi de suma importância ao presente trabalho. À secretária Suzana Gobetti que sempre se dispôs a ajudar em que fosse necessário, acima de tudo como uma grande amiga.

Agradeço ao Centro Nacional de Supercomputação (CESUP) da Universidade Federal do Rio Grande do Sul por disponibilizar o acesso ao *cluster* Gauss, o qual foi de suma importância ao presente trabalho.

Agradeço às instituições de apoio à pesquisa pelo auxílio financeiro com a bolsa de estudos e para participação em eventos, sem as quais seria impossível minha dedicação integral ao mestrado: CNPq, CAPES, INCT-FBN e PRPPG.

Agradeço aos colegas do programa de mestrado das turmas de 2010, 2011 e 2012 pela amizade e companheirismo. Notadamente à Kátia de Paiva Lopes, que esteve em todas as etapas do desenvolvimento deste, tanto como profissional como amiga.

À Tatiana Machado de Souza pelo apoio, incentivo companheirismo e amizade. E, por fim, à Bárbara Nobrega Rodrigues, que sendo muito mais do que amiga e companheira, esteve ao meu lado esses dois anos, sendo uma valiosa fonte de inspiração tanto para minha vida pessoal como profissional.

RESUMO

Transcriptoma é o conjunto completo de transcritos de uma célula em uma dada condição fisiológica e/ou de desenvolvimento. Atualmente, há várias abordagens para se estudar o transcriptoma, tal como a tecnologia de RNA-Seq, a qual é uma metodologia de alta produção (*high-throughput*), alta resolução e baixo custo. Esta tecnologia é inovadora devido à independência do conhecimento prévio do genoma do organismo em estudo, revela os limites de transcrição e variações na sequência, detecta níveis de expressão e não demanda uma grande quantidade de amostras de RNA. Uma das análises principais em dados de RNA-Seq é a detecção de genes diferencialmente expressos em diferentes condições experimentais. O presente trabalho descreve Wedring, um *pipeline* desenvolvido nas linguagens de programação Python e R, além de integrar *softwares* de bioinformática. O objetivo de Wedring é determinar genes diferencialmente expressos a partir de experimentos de RNA-Seq. Wedring também visa ser facilmente inserido em outros *pipelines*, possibilitar o desenvolvimento de serviços *web* para análise de RNA-Seq e prover meios para facilitar a visualização dos dados para os usuários. Wedring usa o *software* Bowtie para mapear leituras curtas em genomas. Bowtie é eficiente em termos de uso de memória e tempo de execução. Wedring também usa o *software* mapeador TopHat, o qual é baseado no Bowtie e é mais adequado para a análise de genomas eucarióticos. Outros *softwares* utilizados por Wedring são SAMtools e BEDTools, que são utilizados para manipular arquivos de mapeamento no formato SAM/BAM. O último *software* utilizado por Wedring é o DESeq, uma biblioteca de R do projeto Bioconductor. DESeq visa aplicar estatísticas sobre contagens para se inferir genes diferencialmente expressos dado um número de condições experimentais. Os dados de entrada para o Wedring são o genoma de referência, o arquivo de características genômicas no formato GFF, as bibliotecas e as condições experimentais. Dados de RNA-Seq de *Azospirillum brasilense* fp2, uma bactéria fixadora de nitrogênio, foram utilizados para se testar o Wedring. Um total de duas bibliotecas controle e três de tratamento (condição de fixação de *A. brasilense* fp2), sequenciadas utilizando-se a tecnologia SOLiD foram utilizadas. Ao final da análise 95 genes foram indicados como diferencialmente expressos. Uma análise de qualidade das bibliotecas foi feita e os parâmetros de Wedring foram redefinidos para remover 20 bases da extremidade 3’ das leituras. Essa mudança fez com que Bowtie se tornasse mais seletivo para reportar mapeamentos válidos, resultando em 141 genes diferencialmente expressos (67 sobreexpressos e 74 subexpressos). A análise de *A. brasilense* fp2 com o Wedring foi realizada utilizando-se uma única linha de comando, facilitando o processo de análise de expressão gênica em experimentos de RNA-Seq.

Palavras-chave: Wedring, RNA-Seq. Expressão gênica diferencial. *Azospirillum brasilense* fp2. Fixação biológica de nitrogênio.

ABSTRACT

Transcriptome refers to the complete set of the RNA transcripts for a specific cellular developmental/physiological condition. Currently, there are several approaches to study the transcriptome, such as the sequence-based ones, like RNA-Seq technology. RNA-Seq is a recent high-throughput, high resolution and low cost methodology. This technique is innovative due to its independence of previous knowledge about the genome of the target organism. It reveals transcription boundaries, sequence variations, detects expression levels and doesn't require large amounts of RNA samples. One of the main analysis using RNA-Seq data is the detection of differentially expressed genes across different experimental conditions. The current work describes Wedring, a pipeline developed using Python and R programming languages and bioinformatical softwares, whose objective is the achievement of differentially expressed genes lists derived from RNA-Seq experiments. It also aims to be easily embedded in other pipelines due to its modular nature, enable the development of web-services for RNA-Seq analysis and provide users with an easy vizualization of data. Wedring pipeline uses the Bowtie aligner to map short reads to genomes, this software is efficient in terms of memory footprint and execution time. Bowtie indexes genomes using the Burrows-Wheeler Transformation to achieve fast searches in the genome using low amounts of computer memory. Wedring also uses the TopHat mapper (which has Bowtie as its base). Because TopHat is designed to discover splicing junctions, it is more suitable to analyze eukaryotic genomes than Bowtie alone. Other tools used by Wedring are SAMtools and BEDTools, which are softwares used to manipulate mapping files in SAM/BAM formats. The last software used by Wedring is the DESeq, an R library from the Bioconductor project. DESeq aims to apply statistics based on the Negative Binomial Distribution over a count dataset to discover which genes are differentially expressed given a number of experimental conditions. Wedring's inputs are the reference genome, the genomic features file in GFF format, the sequenced libraries and the experimental conditions. RNA-Seq data of *Azospirillum brasilense* fp2, a nitrogen fixing bacteria, was used as a study case to test Wedring. A total of five libraries, two controls and three treatments, sequenced with SOLiD were used, where the treatments refer to the fixing condition of *A. brasilense* fp2. At the end of the analysis 95 genes were differentially expressed. A quality analysis of the libraries was performed and Wedring's parameters were reset to trim 20 bases from the reads 3' end. With this change Bowtie could be more selective to report a valid mapping, resulting in 141 differentially expressed genes (67 upregulated, 74 downregulated). The analysis of *A. brasilense* fp2 with Wedring was performed using a single command line, easing the process of analyzing gene expression in RNA-Seq experiments.

Key-words: Wedring. RNA-Seq. Differential Gene Expression. *Azospirillum brasilense* fp2. Biological Nitrogen Fixation.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 TRANSCRIPTOMA

Transcriptoma é o conjunto de todos transcritos, tal como suas quantidades, que compõe uma célula em um dado estágio de desenvolvimento ou condição fisiológica. Entender o transcriptoma é essencial para interpretar os elementos funcionais de um genoma, revelar componentes moleculares de células e tecidos e entender o desenvolvimento e doenças (MARTIN; WANG, 2011; WANG; GERSTEIN; SNYDER, 2009).

As metas do transcriptoma são: (1) catalogar todos os tipos de transcritos (mRNA, ncRNA, sRNA etc.); (2) determinar a estrutura transcricional do gene em termos de seus sítios de início e fim (extremidades 5’ e 3’ respectivamente), padrões de *splicing* e outras modificações pós-transcricionais; e (3) quantificar os níveis de expressão de cada transcrito durante o desenvolvimento e sob diferentes condições (BERGER *et al.*, 2010; WANG; GERSTEIN; SNYDER, 2009).

Existem várias técnicas para deduzir e quantificar o transcriptoma de um organismo, das quais se inclui as que são baseadas em hibridização e as baseadas em sequênciamento de RNA.

1.1.1 TÉCNICAS BASEADAS EM HIBRIDIZAÇÃO

As técnicas baseadas em hibridização foram as primeiras a serem utilizadas para se realizar estudos em transcriptômica (OZSOLAK; MILOS, 2011).

Essas técnicas se baseiam ligação complementar de amostras de mRNA a moléculas de DNA com sequências de genes previamente conhecidas fixadas em um arranjo pré-definido, chamado de DNA *microarray* ou DNA *chip*. As amostras de

mRNA são utilizadas na forma de cDNA preparadas utilizando-se nucleotídeos marcados com fluorocromos, sendo que para cada amostra é utilizada um flurocromo de cor específica (LEE, N. H.; SAEED, 2007; TANIGUCHI et al., 2001).

Após um período de incubação para que a reação de hibridização seja efetivada, o *chip* é escanerizado usando-se tecnologia de laser (para excitar os fluorocromos) e, então, uma imagem digital é gerada e armazenada em um computador para análises posteriores. Essa imagem contem vários pontos, que representam a reação específica de complementariedade entre o cDNA e as sequências de DNA conhecidas. Além disso, a intensidade das cores dos pontos indica o nível de expressão gênica dos genes na amostra e quais genes estão presentes em cada amostra (LEE, N. H.; SAEED, 2007; TANIGUCHI et al., 2001).

A vantagem da utilização de técnicas baseadas em hibridização é a verificação simultânea da expressão de milhares de genes, de uma maneira semiquantitativa e diretamente visualizável (ALBA et al., 2004; LEE, N. H.; SAEED, 2007; YANG et al., 2002). Entretanto, essas técnicas possuem algumas deficiências. É necessário o conhecimento prévio do genoma de referência, a quantidade de RNA amostrado necessita ser alta (ALBA *et al.*, 2004).

Além disso, a comparação entre amostras pode possuir tendenciosidades induzidas pelos fluorocromos utilizados, que podem ser devidas a propriedades físicas desses (sensibilidade ao calor e a luz, meia-vida relativa), eficiência da incorporação e variabilidade experimental no processo de hibridização e de escanerização do *chip* (YANG *et al.*, 2002). Vale ressaltar a existência da hibridização cruzada, na qual o cDNA da amostra hibridiza com sequências semelhantes de DNA fixado no *chip* (OKONIEWSKI; MILLER, 2006; ROYCE; ROZOWSKY; GERSTEIN, 2007).

1.1.2 TÉCNICAS BASEADAS EM SEQUÊNCIA

As técnicas baseadas em sequência possuem uma característica que basicamente as diferenciam das técnicas baseadas em hibridização, que é a

determinação direta da sequência do cDNA. A maior parte dessas técnicas utiliza-se do sequenciamento Sanger (SANGER; NICKLEN; COULSON, 1977), sendo conhecidas como técnicas de "*Expression Sequence Tags*" (EST) (WANG, Z.; GERSTEIN, M.; SNYDER, 2009).

A contrução da biblioteca (ADAMS *et al.*, 1991; PARKINSON; BLAXTER, 2009) de EST's inicia-se com a extração e purificação do mRNA, o qual é convertido em cDNA. As moléculas de cDNA são amplificadas e, então, sequenciadas. Como resultado do sequenciamento são geradas leituras cujo tamanho varia de 200 a 900 pares de bases (ALBA *et al.*, 2004).

EST's proveem um recurso robusto para análise de sequências para a descoberta de genes, anotação de genomas e genômica comparativa (ALBA *et al.*, 2004). Contudo, o uso desse tipo de tecnologia é caro, de baixa produção, não quantitativo, muitas das leituras geradas não são mapeáveis unicamente no genoma de referência e podem não representar o conjunto completo de transcritos do organismo do qual o mRNA foi originado (por exemplo em caso de transcritos pouco abundantes) (ALBA *et al.*, 2004; WANG; GERSTEIN; SNYDER, 2009).

Por outro lado, o desenvolvimento de novas tecnologias de sequenciamento de DNA proveu um novo método que viabiliza o mapeamento e quantificação do transcriptoma, chamado RNA-Seq (WANG, Z; GERSTEIN; SNYDER, 2009).

1.2 SEQUENCIAMENTO DE RNA (RNA-SEQ)

O desenvolvimento da tecnologia do RNA-Seq proporcionou ao estudo do transcriptoma um avanço com relação às técnicas utilizadas anteriormente (OSHLACK; ROBINSON, M. D.; YOUNG, 2010). Esse avanço está diretamente ligado ao surgimento das tecnologias *high-throughput* (alta produção) de sequenciamento, pois essas permitiram que o sequenciamento de RNA pudesse ser realizado em larga escala a um custo relativamente baixo (OZSOLAK; MILOS, 2011).

RNA-Seq proporciona as seguintes vantagens: (1) não depende do conhecimento prévio do genoma do organismo em estudo; (2) revela localização

precisa dos limites de transcrição; (3) revela variações nas sequências com precisão de uma base; (4) sensível em detectar diversos níveis de expressão; e (5) não necessita de muitas amostras de RNA para se realizar o experimento (PARKHOMCHUK *et al.*, 2009; WANG; GERSTEIN; SNYDER, 2009).

Além disso, a TABELA 1 apresenta uma comparação entre as tecnologias de *microarray*, EST e RNA-Seq, enfatizando as vantagens dessa última em relação as outras.

TABELA 1 - TECNOLOGIAS EM TRANSCIPTÔMICA. Comparação entre as tecnologias de RNA-SEQ, *microarray* e EST em termos de princípio de sequenciamento, resolução, produção (*throughput*), uso de referência, ruído, quantidade de amostra e custo.

ESPECIFICAÇÕES	TECNOLOGIA		
	MICROARRAY	*EST*	*RNA-SEQ*
Princípio	Hibridização	Sanger	*High-throughput*
Resolução	Até 100 bases	Base única	Base única
Produção	Alta	Baixa	Alta
Uso de referência	Sim	Não	Alguns casos
Ruído	Alto	Baixo	Baixo
Quantidade de amostra	Alto	Alto	Baixo
Custo	Alto	Alto	Relativamente baixo

FONTE: Modificado de Wang, Gerstein e Snyder (2009).

A FIGURA 1 descreve um experimento típico de RNA-Seq. Nesse experimento uma população total ou fracionada de RNA é convertida em uma biblioteca de fragmentos de cDNA com adaptadores ligados em uma ou ambas as extremidades do fragmento. De acordo com o tipo de protocolo utilizado para o sequenciamento esses fragmentos podem ou não passar por uma etapa de amplificação.

Em seguida, cada molécula é sequenciada em alguma plataforma de sequenciamento *high-throughput*, como *Illumina GA*[1], *Life Technologies SOLiD*[2] (ANEXO 1) e *Roche 454 Life Sciences*[3], gerando centenas de milhares a centenas de milhões de sequencias curtas, chamadas de leituras (MARGUERAT; BÄHLER, 2010; MARTIN; WANG, 2011; WANG; GERSTEIN; SNYDER, 2009). O sequenciamento pode ser feito de tal modo que uma ou ambas as extremidades dos

[1] http://www.illumina.com/technology/sequencing_technology.ilmn
[2] http://www.lifetechnologies.com/global/en/website-overview/ab-welcome.html
[3] http://www.454.com/

fragmentos seja sequenciada; no primeiro caso o sequenciamento é chamado *single-end* e no segundo de *paired-end* (HOLT; JONES, 2008; METZKER, 2010; WANG; GERSTEIN; SNYDER, 2009). As leituras geradas pelo sequenciamento possuem um comprimento de 35-500 pares de bases dependendo da tecnologia utilizada (MARGUERAT; BÄHLER, 2010; METZKER, 2010).

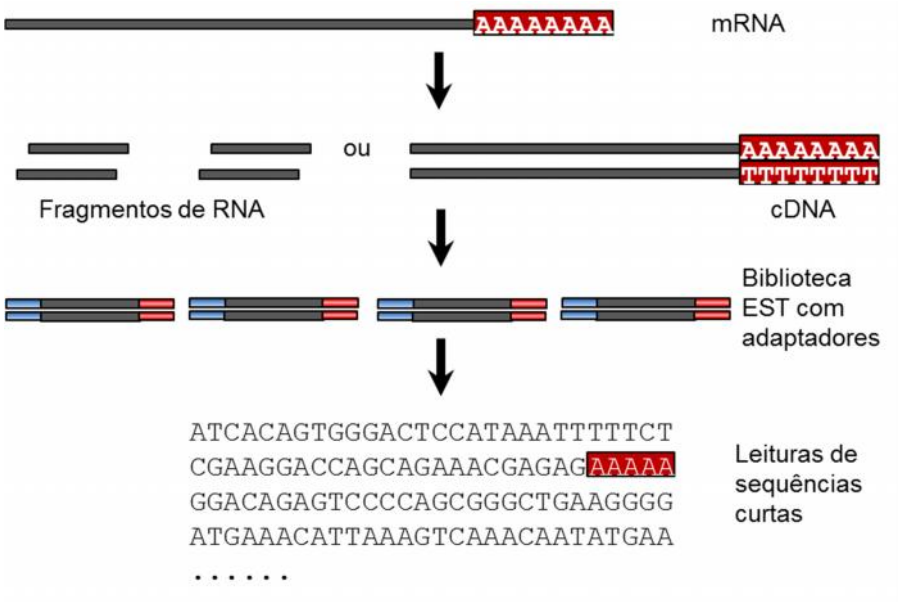


FIGURA 1 – EXPERIMENTO TÍPICO DE RNA-SEQ. Após a extração, o mRNA é convertido para cDNA e ligado a adaptadores. Esse material é processado por uma plataforma de sequenciamento e são geradas milhões de leituras curtas. FONTE: Modificado de Oshlack, Robinson e Young (2010).

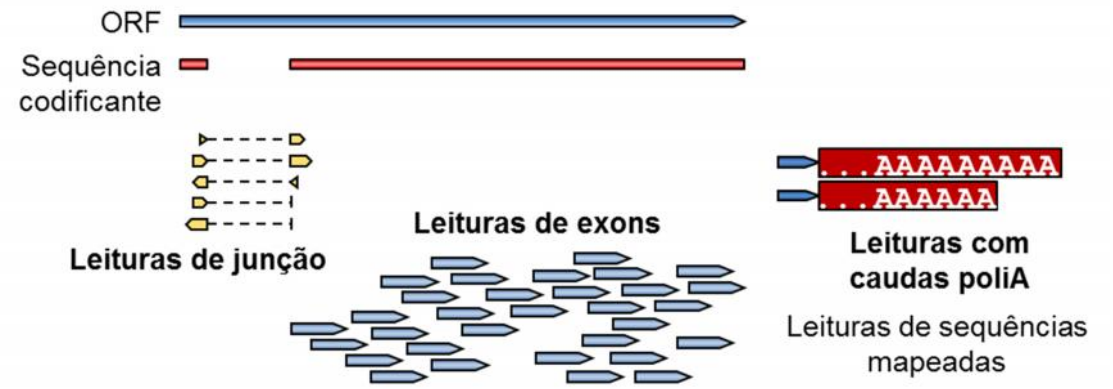


FIGURA 2 – MAPEAMENTO DAS LEITURAS APÓS O SEQUÊNCIAMENTO. Após o sequenciamento, as leituras são mapeadas contra um genoma de referência, empilhando-se nas regiões da onde provavelmente foram originadas. Vale notar que algumas leituras mapeiam em bordas de éxons. FONTE: Modificado de Oshlack, Robinson e Young (2010).

Após o sequenciamento (FIGURA 2) as leituras podem ser processadas de

duas maneiras: (1) mapeamento contra um genoma de referência ou contra transcritos de referência; (2) montagem de novo sem a utilização de genoma de referência (WANG; GERSTEIN; SNYDER, 2009). Adquire-se, assim, um mapa de transcrição em escala genômica que consiste na estrutura transcricional e/ou o nível de expressão de cada gene (NAGALAKSHMI *et al.*, 2008; WANG; GERSTEIN; SNYDER, 2009).

Diversos estudos já foram realizados usando *RNA-Seq* como ferramenta para elucidar fenômenos biológicos. Gregg *et al.* (2010) analisaram o efeito do *imprinting* genômico pela diferença de expressão alelo específica de sítios de SNP em amostras de tecido cerebral de ratos. Berger *et al.* (2010) identificaram fusão de genes e outras mutações em amostras de câncer por meio da descoberta de transcritos novos. Picardi *et al.* (2010) analisaram sítios de edição de RNA em *Vitis vinifera*, identificando 401 sítios de conversão de C para U em RNAs codificantes e 44 modificações em RNAs não codificantes. Também se utilizando *V. vinifera* como modelo, Denoeud *et al.* (2008) desenvolveram uma abordagem para construir modelos gênicos a partir de dados de RNA-Seq, com a qual encontraram 675 modelos novos. Wang *et al.* (2008) relataram a partir do estudo do transcriptoma de 15 tecidos e linhagens celulares humanas que 92-94% dos genes humanos passam pelo processo de *splicing* alternativo. Lawley, Sims e Tannock (2013) estudaram o padrão de expressão gênica de *Lactobacillus ruminis* presente em intestino humano com relação à utilização de tetrassacarídeos liberados da hidrólise de β-glucanos.

Outro método para a análise de dados de RNA-Seq é o que se refere ao perfil de expressão gênica entre amostras. Esse tema será discutido em mais detalhes, devido a sua importância para o presente trabalho.

1.3 ANÁLISE DE EXPRESSÃO DIFERENCIAL EM RNA-SEQ

A análise de expressão gênica diferencial em experimentos de RNA-Seq é relevante em experimentos controlados, nos quais se deseja verificar o efeito de um tratamento sobre a expressão gênica de tecidos e linhagens celulares de um

organismo.

Esses experimentos são delineados utilizando-se ao menos duas amostras, das quais uma será o controle e a outra o tratamento, podendo haver mais de uma réplica para o controle tal como mais de uma para o tratamento. Vale ressaltar que a utilização de réplicas aumenta a acurácia e garante a reprodutibilidade do experimento, uma vez que permite considerar a variação dentro dos tratamentos, além da variação entre tratamentos (AUER; DOERGE, 2010).

A FIGURA 3 sumariza um fluxo de trabalho, ou seja, um pipeline, generalizado sobre a análise de expressão diferencial com dados de RNA-Seq. Ele possui uma série de etapas bem definidas que necessitam ser coordenadas para que os dados sejam processados e interpretações biológicas possam ser realizadas.

Segundo Oshlack, Robinson e Young (2010), as etapas da análise podem ser sumarizadas da seguinte maneira:

Mapeamento. Refere-se ao posicionamento de milhões de leituras em um genoma de referência em trechos cujas sequências sejam idênticas as das leituras, isto é, encontrar a verdadeira posição de origem das leituras. Contudo, esse posicionamento muitas vezes não é exato, sendo necessário considerar não correspondências, ou seja, bases diferentes entre a leitura e o genoma de referência. Assim, o software mapeador deve ser capaz de considerar erros e variações estruturais para que o mapeamento seja o mais próximo possível da realidade (FONSECA *et al.*, 2012).

Sumarização. Consiste na agregação das leituras em unidades biologicamente significativas, como genes, éxons ou transcritos. A maneira mais simples e mais comum de se fazer essa sumarização é a contagem das leituras que sobrepõem os éxons em um gene (BULLARD *et al.*, 2010; MARIONI *et al.*, 2008). Assim, a contagem é considerada como sendo uma estimativa direta da expressão de um dado gene (ANDERS; HUBER, 2010).

Normalização. Essa etapa permite que comparações de nível de expressão dentro de amostras ou entre amostras sejam feitas de maneira mais acurada, devido a retirada de tendenciosidades inerentes as amostras. Para comparações dentro de amostras a normalização pode ser feita dividindo-se o número de leituras sumarizadas pelo tamanho do gene, uma vez que genes maiores tendem a ter mais

leituras mapeadas em relação a genes menores mesmo em um nível semelhante de expressão. Para comparações entre amostras, a normalização é feita pelo número de leituras presentes na biblioteca, uma vez que bibliotecas maiores tendem a ter mais leituras mapeadas no genoma (ROBINSON, M. D.; SMYTH, 2007).

Expressão diferencial: Essa etapa visa realçar diferenças no nível de expressão entre as condições experimentais analisadas. Para tanto se aplicam testes estatísticos sobre os dados de contagem sumarizados de cada biblioteca. Como dados de contagem são variáveis discretas, algumas distribuições de probabilidade discretas são mais adequadas para se extrair informações dos dados de RNA-Seq, como a distribuição de Poisson (MARIONI *et al.*, 2008) e a distribuição Binomial Negativa (ANDERS; HUBER, 2010).

De acordo com a lógica dessas etapas, primeiramente as leituras originadas do sequenciamento são mapeadas contra um genoma ou um transcriptoma de referência. Então, as leituras mapeadas são sumarizadas de acordo com as características que o estudo em questão esteja analisando. Essas características dependem do nível de abordagem desejado, podendo se referir ao nível de genes, éxons ou junções, por exemplo. Assim, uma tabela de contagens pode ser gerada contendo as contagens de cada característica sumarizada para cada biblioteca sequenciada. Em seguida, os dados são normalizados e testados estatisticamente para que a expressão diferencial seja inferida, produzindo uma lista de características associada com valores de p e de *fold change*[4] para cada uma delas (OSHLACK; ROBINSON; YOUNG, 2010; WILHELM; LANDRY, 2009).

Desse modo, tendo-se em mãos a lista contendo os genes diferencialmente expressos, abordagens de biologia de sistemas podem ser utilizadas para que interpretações biológicas possam ser realizadas sobre os dados dos experimentos de RNA-Seq (OSHLACK; ROBINSON; YOUNG, 2010). Para tanto, os gene são agrupados em categorias que compartilhem propriedades biológicas em comum e, então, essas categorias são testadas estatisticamente para que se verifique qual delas está representada diferencialmente entre os genes diferencialmente expressos (YOUNG *et al.*, 2010). Atualmente, os vocabulários do Gene Ontology (GO) (THE GENE ONTOLOGY CONSORTIUM *et al.*, 2011) são comumente utilizados para se

[4] Descreve o quanto uma medida muda de um estado inicial para um estado final. No caso refere-se à mudança de expressão gênica entre condições experimentais.

aplicar essa técnica (HUANG, D. W. *et al.*, 2007).

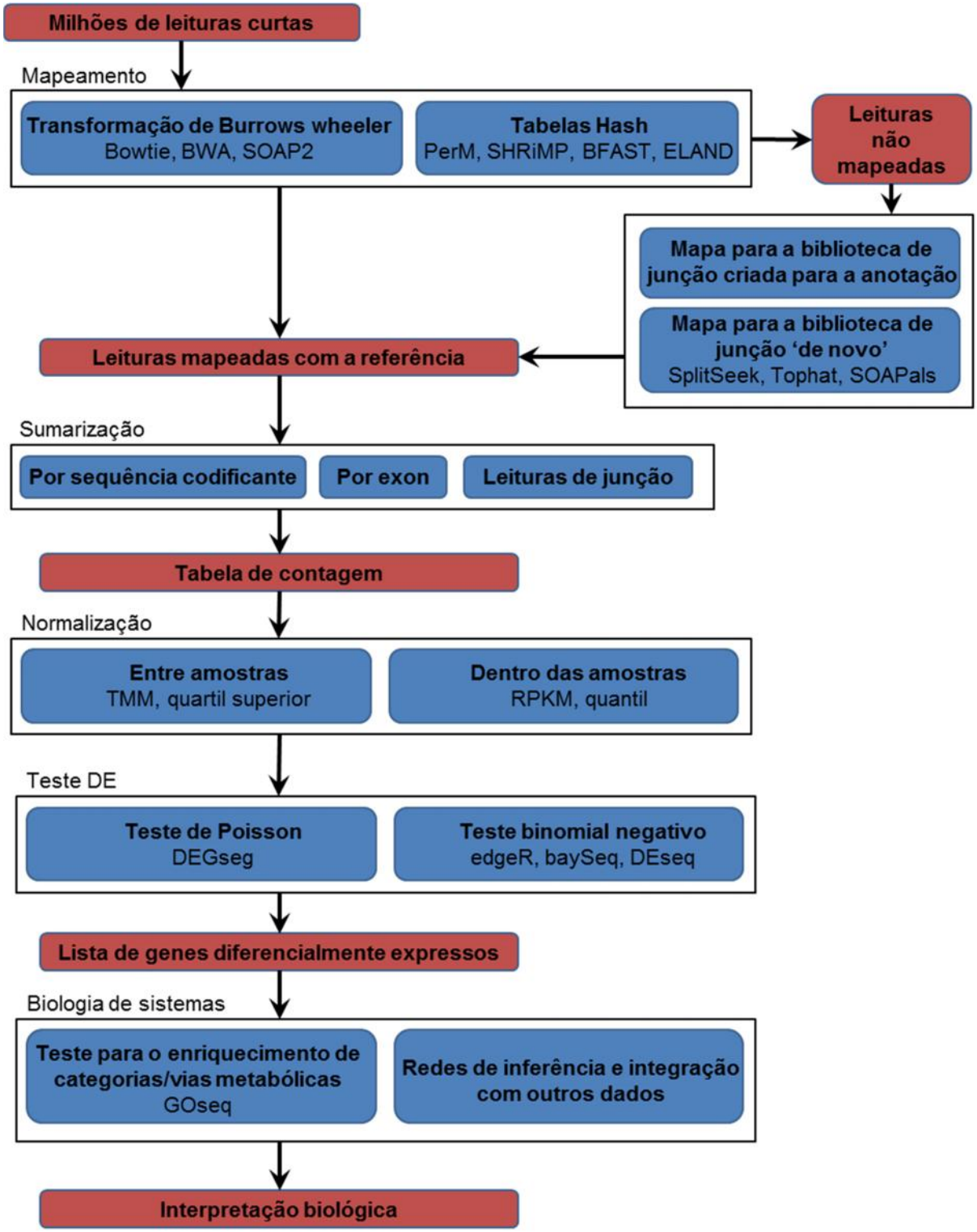


FIGURA 3 – VISÃO GERAL SOBRE UM *PIPELINE* DE ANÁLISE DE EXPRESSÃO GÊNICA EM EXPERIMENTOS DE RNA-SEQ. Após o sequenciamento, as leituras são mapeadas contra um genoma de referência e sumarizadas para gerar tabelas de contagem. Após a etapa de normalização, estatísticas de expressão diferencial podem ser aplicadas. A partir da lista de genes diferencialmente expressos, análises de biologia de sistemas podem ser aplicadas para que interpretações biológicas possam ser realizadas sobre os dados. FONTE: Modificado de Oshlack, Robinson e Young (2010).

Como um exemplo do fluxo de trabalho que pode ser realizado com dados oriundos de experimentos de RNA-Seq, pode-se citar Camarena *et al.*, (2010). Nesse estudo, foi avaliado o perfil de expressão gênica entre amostras do patógeno *Acinetobacter baumannii* ATCC17978[5] sob o efeito de etanol. Essa cepa bacteriana foi cultivada em dois tipos de meios de cultura, um com a adição de etanol e outro sem adição desse álcool. A partir dessas culturas, quatro bibliotecas de cDNA foram preparadas: dois controles (sem etanol) e dois tratamentos (com etanol). As bibliotecas foram sequenciadas utilizando-se a plataforma Illumina e foram geradas 6.441.146 leituras para os controles e 6.603.654 para os tratamentos. Essas leituras foram mapeadas contra o genoma de referências dessa espécie. Analisando-se o número de leituras mapeadas em cada região codificante (comparando bibliotecas de controle com as de tratamento), verificou-se que o etanol induziu a expressão de 49 genes, pertencentes a diversas categorias funcionais, como as de estresse celular, permeabilidade celular, assimilação de ferro, entre outras.

A análise de expressão diferencial em RNA-Seq necessita lidar com uma enorme quantidade de dados gerada pelo sequenciamento e, devido a complexidade desses dados, sua interpretação não é direta, além de exigir intensamente recursos computacionais (por exemplo, memória RAM) (FONSECA *et al.*, 2012; MARGUERAT; BÄHLER, 2010; METZKER, 2010; OSHLACK; ROBINSON; YOUNG, 2010). Portanto, em todas as etapas da análise, os dados utilizados necessitam ser processadas por meios computacionais para que se possam realizar inferências biológicas o mais acuradas possível acerca desses dados (OSHLACK; ROBINSON, M. D.; YOUNG, 2010).

Diversos softwares podem ser utilizados em pipelines de ánalise de expressão diferencial em experimentos de RNA-Seq (FONSECA *et al.*, 2012; GAO *et al.*, 2010; OSHLACK; ROBINSON; YOUNG, 2010; YENDREK; AINSWORTH; THIMMAPURAM, 2012). Essas ferramentas utilizam-se de arquivos como dados de entrada (arquivos de leituras, genoma de referência, anotação de genoma, tabelas, entre outros) e geram arquivos como dados de saída (arquivos de mapeamento, tabelas, entre outros).

Entretanto, não existe padronização oficial para definir diretivas de

[5] http://www.ncbi.nlm.nih.gov/genome/?term=Acinetobacter%20baumannii%20ATCC17978

formatação para esses arquivos. Alguns esforços para a definição de especificações e práticas recomendadas sobre o uso desses arquivos já foram realizados pela comunidade científica. Por exemplo, pode-se citar o caso do arquivo de anotação no formato GFF (DURBIN; HAUSSLER, David, 2009), o arquivo para armazenamento de leituras Fastq (COCK *et al.*, 2010) e os arquivos de mapeamento SAM e BAM (THE SAM FORMAT SPECIFICATION WORKING GROUP, 2011).

Ademais, há o caso do formato Fasta, o qual é um tipo de arquivo amplamente utilizado em bioinformática, mas que não possui nenhum tipo de especificação disponível. Porém, esforços no sentido de viabilizar uma especificação para esse formato encontram-se em andamento, como o realizado pela Universidade de Michigan no projeto ProteomeCommons.org (FALKNER; HILL; ANDREWS, 2008; FALKNER; ULINTZ, 2013).

Apesar de ainda não haverem padronizações oficiais para os formatos de arquivos utilizados na análise de expressão diferencial de RNA-Seq, a presença de esforços para viabilizar especificações torna esses formatos mais adequados para a utilização em pipelines, uma vez que um fluxo de informação bem estruturado entre *softwares* é crucial para a análise.

1.3.2 FORMATOS DE ARQUIVO

1.3.1.1 FASTA

Fasta é um formato de arquivo de texto, no qual se representam sequências biológicas. O formato Fasta foi inventado por Pearson e Lipman (1988) para ser utilizado em um software de alinhamento de sequências chamado FASTA.

O formato é composto por um cabeçalho, descritivo sobre a que se refere a sequência, seguido da sequência (de nucleotídeos ou de aminoácidos) em si nas linhas subsequentes. O cabeçalho é sempre iniciado pelo caractere maior-que (“>”)

posicionado na primeira coluna do arquivo para diferenciá-lo das linhas de sequência. Várias sequências podem estar presentes em um mesmo arquivo, nesse caso o arquivo é chamado de Multifasta. É recomendado que nenhuma linha desse arquivo ultrapasse 80 caracteres (FALKNER; ULINTZ, 2013; NCBI, 2013). Na FIGURA 4A encontra-se um exemplo do formato Fasta.

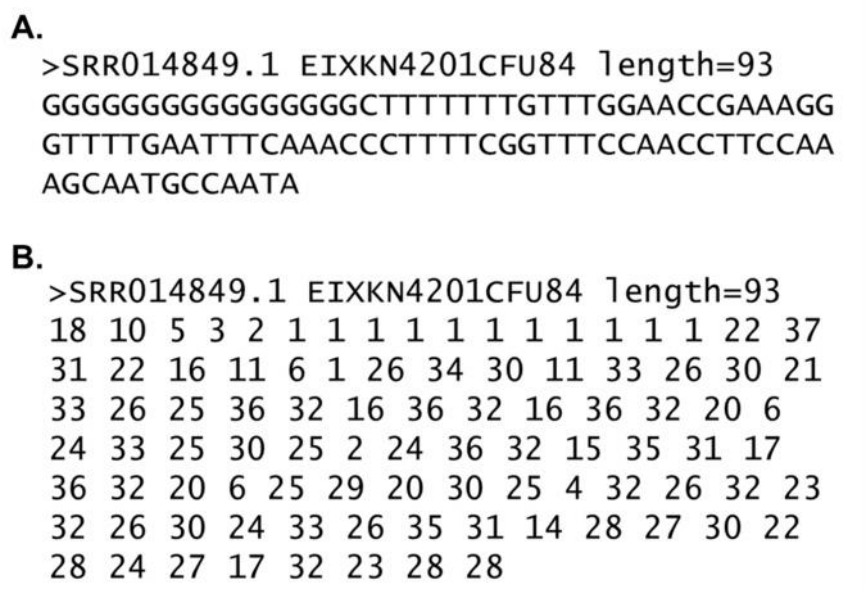


FIGURA 4 – SEQUÊNCIA EM FORMATO FASTA E VALORES DE QUALIDADE PHRED NO FORMATO QUAL. (A) Sequência nucleotídica no formato Fasta. Notar o caractere “>” marcando o início do cabeçalho do registro. (B) Valores de qualidade PHRED da sequência apresentada em (A). FONTE: Modificado de Cock *et al.* (2010).

1.3.1.2 QUAL

O formato Qual é estruturalmente idêntico ao formato Fasta, mas representa valores de qualidade específicos para cada base da sequência ao invés da sequência em si. Esse formato foi introduzido por um software chamado PHRED (EWING *et al.*, 1998; EWING; GREEN, 1998), o qual analisa os dados resultantes de sequenciamento Sanger (SANGER; NICKLEN; COULSON, 1977) para definir as bases da sequência (*base call*) e atribuir um valor de qualidade para cada base. O valor de qualidade é estimado como

$$Q_{PHRED} = -10 \times log_{10}(P_e), \quad (1)$$

sendo P_e a probabilidade da base estar errada (EWING; GREEN, 1998). Assim, para todas as sequências (as quais são armazenadas no arquivo Fasta) os valores de qualidade são gerados e armazenados no arquivo Qual como valores inteiros separados por espaço. Diferentes plataformas de sequenciamento podem apresentar maneiras distintas de se estimar os valores de qualidade (TABELA 2), contudo todas elas utilizam os princípios semelhantes aos do software PHRED (COCK *et al.*, 2010). Na FIGURA 4B pode-se ver um exemplo do formato Qual.

1.3.1.3 FASTQ

Fastq é um formato de arquivo de texto utilizado para representar sequências geradas pelo sequenciamento realizado por diversas plataformas, como Illumina GA e Life Technologies SOLiD (COCK *et al.*, 2010).

Nesse formato, a informação da sequência e da qualidade das leituras é armazenada em um mesmo arquivo. Um exemplo de leitura no formato Fastq pode ser encontrado na FIGURA 5.

```
@SRR014849.1 EIXKN4201CFU84 length=93
GGGGGGGGGGGGGGGCTTTTTTTGTTTGGAACCGAAAGG
GTTTTGAATTTCAAACCCTTTTCGGTTTCCAACCTTCCAA
AGCAATGCCAATA
+SRR014849.1 EIXKN4201CFU84 length=93
3+&$#""""""""""""7F@71,'";C?,B;?6B;:EA1EA
1EA5'9B:?:#9EA0D@2EA5':>5?:%A;A8A;?9B;D@
/=<?7=9<2A8==
```

FIGURA 5 – LEITURA REPRESENTADA NO FORMATO FASTQ. Em vermelho, os caracteres "@", mostrando o início do cabeçalho, e "+", a divisão entre a sequência e os valores de qualidade. FONTE: Modificado de Cock *et al.* (2010).

Cada registro de um arquivo Fastq é composto por quatro linhas (COCK *et al.*, 2010).

Cabeçalho. Iniciada com o caractere arroba (“@”), essa linha não possui limite de comprimento e contém informações que visam identificar singularmente cada leitura.

Sequência. Essa linha contém a sequência da leitura, geralmente representada em letras maiúsculas, embora minúsculas também possam ser usadas. Espaços em branco e tabulações não são permitidos e sequência pode ser dividida em várias linhas.

Separador. Essa linha contém um caractere mais (“+”), que representa o final da sequência e início da linha de qualidade. O caractere “+” pode ou não vir seguido do mesmo conteúdo do cabeçalho.

Qualidades. Última linha do registro contém a informação sobre a qualidade da leitura, que também pode ser dividida em várias linhas. Os valores de qualidade são representados por subconjuntos de caracteres imprimíveis da tabela ASCII (AMERICAN NATIONAL STANDARD FOR INFORMATION SYSTEMS, 1986), sendo que o intervalo de valores utilizado é específico de cada plataforma de sequenciamento (TABELA 2). A cadeia de caracteres da qualidade deve possuir o mesmo tamanho da sequência.

TABELA 2 – VALORES DE QUALIDADE EM TRÊS VARIANTES DO FORMATO FASTQ. Para cada plataforma é apresentado o intervalo de caracteres ASCII imprimíveis, o tipo de qualidade utilizada e o intervalo de valores de qualidade.

DESCRIÇÃO	CARACTERES ASCII	VALORES DE QUALIDADE	
	INTERVALO	TIPO	INTERVALO
Sanger	33-126	PHRED	0 até 93
Solexa/Illumina	59-126	Solexa	-5 até 62
Illumina 1.3 +	64-126	PHRED	0 até 62

FONTE: Modificado de Cock *et al.* (2010).

1.3.1.4 CSFASTA

Csfasta (*Color Space Fasta*) é um formato de arquivo de texto para representar sequências das leituras geradas pelo sequenciador SOLiD. É um formato semelhante ao Fasta, mas as bases são codificadas em espaço de cor (ANEXO 1). Além do arquivo com as sequências é gerado um arquivo Qual contendo as qualidades das sequências para cada leitura. Existe também o formato Csfastq (COCK *et al.*, 2010), no qual as leituras são armazenadas no formato Fastq, porém as sequências são codificadas em espaço de cor.

1.3.1.5 GFF

O formato GFF (Gene Feature Format) é um formato no qual se descrevem características presentes na anotação de um genoma. Cada característica é descrita em uma linha, que pode possuir de 8 a 9 campos separados por tabulação (DURBIN; HAUSSLER, 2009). A

TABELA 3 descreve cada um desses campos.

TABELA 3 – CAMPOS PRESENTES EM UM ARQUIVO NO FORMATO GFF.

COLUNA	DEFINIÇÃO	DESCRIÇÃO
1	Nome	Nome da sequência de origem da característica (ex.: nome do genoma).
2	Fonte	Nome da fonte de origem da característica (ex.: base de dados, software).
3	Característica	Nome do tipo de característica (ex.: gene, exon, mRNA).
4	Início	Posição de início da característica no genoma de origem. Valor inteiro.
5	Fim	Posição de final da característica no genoma de origem. Valor inteiro.
6*	Pontuação	Pontuação atribuída pelo software anotador. Valor decimal.
7*	Fita	Sentido da leitura da sequência a que a característica corresponde.
8*	Fase	Fase de leitura da característica (aceita os valores 0, 1 ou 2).
9	Atributo	Descrição da característica e outras informações. Opcional.

* Quando a informação desses campos não existe ou não é relevante utiliza-se o caractere ponto (“.”).
FONTE: Modificado de Durbin e Haussler (2009).

1.3.1.6 SAM/BAM

O formato SAM (*Sequence Alignment/Mapping*) foi criado por Li *et al.* (2009) com o objetivo de padronizar a formatação dos arquivos de saída de softwares de mapeamento. Esse formato é representado por um arquivo de texto, no qual são armazenadas as informações acerca de mapeamentos (FIGURA 6A). São suportadas leituras de bibliotecas *single-end* ou *paired-end* e as sequências podem estar representadas tanto em *basespace* como em *colorspace*. Ademais, mapeamentos contra vários genomas de referência podem ser representados em um mesmo arquivo SAM.

O arquivo é dividido em duas seções, a de cabeçalho e a de mapeamentos. O cabeçalho é uma seção opcional e corresponde as linhas iniciadas com o caractere “@”. Na seção de mapeamentos cada linha corresponde a um mapeamento individual, composto por 11 campos mandatórios separados por tabulação, que precisam sempre aparecer na mesma ordem. Quando um valor para o campo não está disponível, podem-se utilizar os caracteres zero (“0”) ou asterisco (“*”), dependendo do campo.

De acordo com a especificação do formato SAM (THE SAM FORMAT SPECIFICATION WORKING GROUP, 2011), as sequências que são mapeadas contra um genoma de referência são chamadas de moldes. No caso de sequenciamento *paired-end*, cada leitura (molde) do par é chamada de segmento. Os 11 campos mandatórios são os seguintes:

QNAME. Nome da sequência que está sendo mapeada, ou seja, o nome da leitura. A presença de mapeamentos de leituras com o mesmo nome indicam mapeamentos em múltiplos locais no genoma de referência e/ou em vários genomas de referência.

FLAG. Marcação de *bits* utilizada para qualificar o mapeamento. A TABELA 4 apresenta os valores possíveis para esse campo. O valor desse campo é definido pela operação lógica OU *bit* a *bit* sobre um conjunto de valores referentes a um mapeamento específico, produzindo valores únicos (FIGURA 6B). Esse campo ainda

pode ser definido como “0”, significando que o molde foi originado de um sequenciamento *single-end* e que foi mapeado com sucesso no genoma de referência, mas não na fita complementar.

A.

```
Coordenadas 12345678901234  56789012345678901234567890123456789012345
ref         AGCATGTTAGATAA**GATAGCTGTGCTAGTAGGCAGTCAGCGCCAT

+r001               TTAGATAAAGGATA*CTG
+r002                          ATAGCT..............TCAGC
-r001                                                CAGCGCCAT
```

QNAME	FLAG	RNAME	POS	MAPQ	CIGAR	RNEXT	PNEXT	TLEN	SEQ	QUAL
r001	163	ref	7	30	8M2I4M1D3M	=	37	39	TTAGATAAAGGATACTG	*
r002	0	ref	16	30	6M14N5M	*	0	0	ATAGCTTCAGC	*
r001	83	ref	37	30	9M	=	7	-39	CAGCGCCAT	*

B.

```
00000001      1          00000001      1
00000010      2          00000010      2
00100000     32          00010000     16
10000000    128          01000000     64
--------OU  ---+         --------OU  --+
10100011  =  163         01010011  =  83
```

FIGURA 6 – MAPEAMENTO NO FORMATO SAM. (A) Mapeamento de leituras em um genoma de referência hipotético (*ref*) e representação em formato SAM com o nome de cada coluna. (B) Cálculo da marcação *bit* a *bit* do campo FLAG para ambos os pares da leitura r001. FONTE: Modificado de Li *et al.* (2009).

Na FIGURA 6 a leitura r001 pertence a um par e a leitura r002 mapeia em uma junção de éxon. Para a leitura r001 o valor 163 (1+2+32+128) no campo FLAG significa que pertence a um par, sendo o segundo segmento, os dois segmentos do molde foram mapeados e seu par foi mapeado na fita reversa. Essa leitura mapeou na posição 7 do genoma de referência, seu par mapeou na posição 37 e de sua primeira base até a última base de seu par são compreendidas 39 bases. O par da leitura r001 possui informações complementares a essa. O valor dessa leitura no campo FLAG (1+2+16+64=83) indica, além dos valores idênticos ao da outra leitura, que esse segmento corresponde ao primeiro do par e foi mapeado na fita complementar. A leitura r002 mapeou na posição 16 e possui o valor 0 no campo FLAG, porque não é uma leitura pareada.

TABELA 4 – VALORES POSSÍVEIS DO CAMPO FLAG EM UM MAPEAMENTO NO FORMATO SAM.

VALOR		DESCRIÇÃO
HEXADECIMAL	DECIMAL	
0x1	1	Molde tem múltiplos segmentos no sequenciamento (ex.: paired-end).
0x2	2	Ambos os segmentos foram mapeados pelo software mapeador.
0x4	4	Segmento não mapeado.
0x8	8	Próximo segmento do molde não foi mapeado.
0x10	16	Segmento mapeado na fita complementar.
0x20	32	Próximo segmento do molde foi mapeado na fita complementar.
0x40	64	Primeiro segmento do molde.
0x80	128	Segundo segmento do molde.
0x100	256	Mapeamento secundário.
0x200	512	Controles de qualidade sobre o molde falharam.
0x400	1024	PCR ou duplicata ótica.

FONTE: Modificado de The SAM Format Specification Working Group (2011).

RNAME. Nome do genoma de referência. Esse campo assume o valor do caractere “*” quando a leitura não pode ser mapeada no genoma de referência.

POS. Posição do genoma de referência onde o mapeamento ocorreu. Sempre se considera a base mais a esquerda da leitura, mesmo se o mapeamento ocorreu na fita reversa. Para leituras não mapeadas esse campo é definido como “0”.

MAPQ. Valor de qualidade do mapeamento. O cálculo do valor desse campo é semelhante ao da qualidade PHRED (EQUAÇÃO 1), mas considera a probabilidade da posição mapeada estar errada. O valor 255 indica que a qualidade do mapeamento não está disponível.

CIGAR. Código que representa como a leitura foi mapeada no genoma de referência, levando em consideração o que ocorreu em cada base da leitura. Esse código informa, por exemplo, quantas correspondências houve entre a leitura e a referência, tal como não correspondências, inserções, deleções entre outros, na ordem em que esses ocorreram. A TABELA 5 descreve todas as operações que compõem o código CIGAR.

TABELA 5 – OPERAÇÕES DO CÓDIGO CIGAR EM UM MAPEAMENTO NO FORMATO SAM.

OPERAÇÃO	DESCRIÇÃO
M	Correspondência.
I	Inserção na referência.
D	Deleção da referência.
N	Região omitida na referência (exemplo, íntrons).
S	Corte suave (não correspondência).
H	Corte rígido (bases não correspondentes não estão presentes na sequência da leitura).
=	Correspondência completa entre leitura e referência.
X	Não correspondência completa entre leitura e referência.

FONTE: Modificado de THE SAM FORMAT SPECIFICATION WORKING GROUP, 2011.

Na FIGURA 6, de acordo com o código CIGAR, no mapeamento da leitura r001 ocorreram 8 correspondências, 2 deleções, 4 correspondências, 1 deleção e 3 correspondências, respectivamente. Já, para o par da leitura r001, seu mapeamento ocorreu em 9 correspondências. A leitura r002 apresentou em seu mapeamento 6 correspondências, 14 não correspondências (indica mapeamento em borda de éxons) e 5 correspondências, respectivamente.

RNEXT. Nome da sequência de referência na qual o próximo segmento do molde foi mapeado. Quando o nome está indisponível, utiliza-se o caractere “*”; quando seu valor é o identifico ao do campo RNAME, utiliza-se o caractere igual (“=”).

PNEXT. Posição onde o próximo segmento do molde foi mapeado, isto é, o valor do campo POS desse. Quando a posição não está definida utiliza-se “0”.

TLEN. Em um molde que possui dois segmentos, se ambos mapearam na mesma referência, esse campo se refere ao número de bases entre a base mais a esquerda do segmento mapeado mais a esquerda e a base mais a direita do segmento do molde mapeado mais a direita. Esse valor que esse campo recebe é positivo para o segmento mais a esquerda e negativo para o segmento mais a direita, sendo que esse valor é igual em ambos. O valor é definido como “0”, quando um segmento é mapeado ou pela indisponibilidade dessa informação.

SEQ. A sequência de nucleotídeos da leitura. O caractere “*” é utilizado quando a sequência não é armazenada.

QUAL. A sequência dos valores de qualidade da leitura. O caractere “*” é

utilizado quando as qualidades não são armazenadas.

Associado ao formato SAM existe outro formato para o armazenamento de mapeamentos chamado BAM (*Binary Alignment/Mapping*) (LI *et al.*, 2009). Esse formato armazena exatamente as mesmas informações que o arquivo SAM, entretanto, como seu próprio nome sugere, os dados são armazenados em código binário. O objetivo desse formato é proporcionar acesso randômico rápido a mapeamentos armazenados no arquivo BAM que se sobrepõem em uma dada região do genoma de referência (LI *et al.*, 2009). Isso é possível porque os arquivos BAM são comprimidos no formato de compressão BZGF - desenvolvido por (LI *et al.*, 2009) - em combinação com um esquema de indexação de blocos, cujo algoritmo é o mesmo utilizado pelo UCSC Genome Browser[6] (KENT *et al.*, 2002)

1.3.2 SOFTWARES

1.3.2.1 BOWTIE

Bowtie (LANGMEAD *et al.*, 2009) é um *software* utilizado para o mapeamento de grandes quantidades de leituras curtas e longas em genomas de referência. As leituras podem possuir comprimento de até 1024 pares de bases e não há limites estabelecidos para o comprimento do genoma de referência. Esse *software* foi desenvolvido para ser eficiente em termos de uso de memória e tempo de execução.

As leituras utilizadas como dado de entrada para o Bowtie podem estar no formato Fasta, Fastq, Csfasta ou Csfastq. Arquivos no formato Qual também podem ser fornecido junto com os arquivos de leituras associados. Bowtie aceita leituras provindas de sequenciamento *single-end* ou *paired-end* e essas podem estar codificadas tanto em espaço de bases como em espaço de cor (nesse caso utiliza-

[6] http://genome.ucsc.edu/

se o parâmetro *-C/--color*). Como saída Bowtie gera um arquivo SAM com os mapeamentos reportados. Esse mapeador pode fazer uso de múltiplos núcleos de processamento para aumentar a velocidade do processo de mapeamento.

Bowtie é a base para diversos outros softwares utilizados em bioinformática, como TopHat (TRAPNELL; PACHTER; SALZBERG, 2009) e Cufflinks (TRAPNELL *et al.*, 2010).

Antes que a etapa de mapeamento seja realizada, o genoma de referência deve ser indexado utilizando-se o Índice de Burrows-Wheeler. Para a construção do Índice de Burrows-Wheeler o genoma de referência deve ser processado utilizando-se a Transformação de Burrows-Wheeler.

A Transformação de Burrows-Wheeler (BURROWS; WHEELER, 1994) é uma técnica que pode ser aplicada a qualquer tipo de texto, não se limitando a sequências biológicas. Essa técnica permite que buscas rápidas sejam feitas em textos grandes, enquanto se mantém baixa a utilização de memória principal do computador.

A Transformação de Burrows-Wheeler de um texto é realizada da seguinte maneira: anexa-se ao fim do texto um caractere não presente nesse texto e lexicograficamente menor que todos os seus caracteres, por exemplo, cifrão (“$”), chamado de pivô. Em seguida, a matriz de Burrows-Wheeler do texto é construída, sendo que suas linhas correspondem a todas as rotações cíclicas desse texto acrescido do caractere pivô. Então, as linhas são ordenadas lexicograficamente. A Transformação de Burrows-Wheeler desse texto corresponde a sequência de caracteres presentes na coluna mais a direita da matriz de Burrows-Wheeler, possuindo o mesmo comprimento que o texto original (FIGURA 7A).

A Transformação de Burrows-Wheeler possui uma propriedade chamada “*last first mapping*”, pela qual a i-ésima ocorrência de um caractere na última coluna da matriz de Burrows-Wheeler corresponde a ocorrência do mesmo caractere na primeira coluna. Essa propriedade é o fundamento de vários algoritmos que realizam operações sobre o Índice de Burrows-Wheeler. Um exemplo é o algoritmo *UNPERMUTE*, que realizada a operação de inversão da Transformação de Burrows-Wheeler, ou seja, a recuperação do texto original (FIGURA 7B).

FIGURA 7 – TRANSFORMAÇÃO DE BURROWS-WHEELER E ALGORITMOS RELACIONADOS. (A) Matriz de Burrows-Wheeler e transformação para a sequência “acaacg”. (B) Algoritmo UNPERMUT, pelo qual se aplica a propriedade “*last first mapping*” para se recuperar o texto original (linha em vermelho no topo) a partir da Transformação de Burrows-Wheeler (coluna mais a direita em preto). (C) Passos do algoritmo EXACTMATCH para se identificar a sequência “aac” na Transformação de Burrows-Wheeler. FONTE: Extraído de Langmead *et al.* (2009).

A Transformação de Burrows-Wheeler possui a característica de aproximar porções do texto que se iniciam com o mesmo caractere, concomitantemente, aproximando porções de texto que são semelhantes. A capacidade de tornar próximas as redundâncias do texto faz com que um texto transformado por essa técnica possa ser comprimido mais facilmente (FERRAGINA; MANZINI, 2000).

Desse modo, aplicando-se uma técnica de compressão – desenvolvida por Ferragina e Manzini (2000) – sobre um texto que foi processado pela Transformação de Burrows-Wheeler, adquire-se um arquivo binário compactado ao qual se dá o nome de Índice de Burrows-Wheeler.

Bowtie constrói o Índice de Burrows-Wheeler a partir de um genoma de referência utilizando-se uma ferramenta chamada *bowtie-build*. Essa ferramenta recebe como parâmetros um ou mais genomas de referência no formato Fasta e uma cadeia de caracteres contendo um nome para o índice. Desse modo, gera-se um índice formatado em espaço de bases. Caso de leituras que serão utilizadas no

mapeamento estejam codificadas em espaço de cor, é necessário utilizar o parâmetro *-C/--color* para que o índice também seja codificado nesse espaço. O índice pode ser reaproveitado em diferentes análises que utilizem o mesmo genoma de referência e leituras codificadas no mesmo espaço.

Além disso, é possível extrair informações do Índice de Burrows-Wheeler. O Bowtie provê essa extração de informações por meio da ferramenta *bowtie-inspect*, que utiliza o algoritmo *UNPERMUT* (BURROWS; WHEELER, 1994) para reconstruir as sequências de referência a partir do índice. Esse *software* também pode fornecer outras informações além das sequências em si, como o tamanho dessas, seus nomes, entre outros.

Uma vez que o Índice de Burrows-Wheeler esteja construído, o mapeamento das leituras pode ser realizado. Para realizar a busca textual dentro do índice, Bowtie utiliza um algoritmo chamado *EXACTMATCH*.

O algoritmo *EXACTMATCH* (FERRAGINA; MANZINI, 2000) baseia-se na propriedade "*last first mapping*" da Transformação de Burrows-Wheeler. Esse algoritmo é muito eficiente em termos de memória, já que carrega na memória principal do computador somente as porções necessárias do índice, quando elas são requisitadas. Ademais, a eficiência em termos de tempo também é conseguida, devido a diminuição do espaço de busca.

O processo de busca (FIGURA 7C) inicia-se pela reconstrução da matriz de Burrows-Wheeler. Tendo-se em mão uma cadeia de caracteres, um intervalo é procurado na primeira coluna da matriz, onde há correspondência com o último caractere da cadeia. Em seguida, o penúltimo caractere é procurado dentro desse intervalo, sendo que o tamanho desse intervalo pode manter-se o mesmo ou diminuir. Esse processo é repetido por todos os caracteres da cadeia. Se a cadeia de caracteres foi encontrada no texto, o algoritmo retorna a cadeia e sua posição dentro do texto. Caso não seja encontrada, uma busca vazia é retornada.

Entretanto, Bowtie não utiliza o algoritmo *EXACTMATCH* exatamente como descrito por Ferragina e Manzini (2000), uma vez que as leituras podem apresentar não correspondências em relação ao genoma de referência, devido a erros de sequenciamento e/ou diferenças reais entre a leitura e o genoma (LANGMEAD *et al.*, 2009).

Bowtie possui dois modos de mapeamento, “v” e “n”. Ambos os modos utilizam o algoritmo *EXACTMATCH* no início do processo de mapeamento. Entretanto, no modo “v” se estabelece um valor máximo de não correspondências que serão permitidas ao longo da leitura no mapeamento. Nesse modo, os valores de qualidade da leitura são ignorados. Esse valor de não correspondências é definido pelo parâmetro *-v* e aceita valores de 0 a 3.

Por outro lado, no modo “n”, Bowtie define uma região chamada semente, que corresponde as primeiras bases da extremidade 5’ da leitura, devido essas bases terem em média valores mais altos de qualidade (HILLIER *et al.*, 2008). O tamanho da semente é definido pelo parâmetro *-l/--seedlen* e por padrão vale 28. Nesse modo, são considerados o número máximo de não correspondências na região da semente (definido pelo parâmetro *-n/--seedmms*, cujo padrão é 2) e a soma dos valores de qualidade em posições onde ocorrem não correspondências não deve ultrapassar um dado valor (definido pelo parâmetro *-e/--maqerr*, cujo padrão é 70).

Quando o Bowtie encontra uma não correspondência, o *EXACTMATCH* retorna uma cadeia de caracteres vazia. Então, Bowtie retrocede no processo de mapeamento e substitui a base não correspondente por uma base diferente na posição onde o *EXACTMATCH* havia sido interrompido e prossegue com o mapeamento. Porém, existem leituras, principalmente aquelas que possuem baixa qualidade, que induzem Bowtie a realizar retrocessos excessivamente.

Para evitar o retrocesso excessivo, Bowtie utiliza uma técnica chamada de “indexação dupla”. Nessa técnica, dois Índices de Burrows-Wheeler são criados, um contendo a sequência do genoma de referência tal como ela é fornecida ao software e outro com a sequência invertida. Um mapeamento válido pode ocorrer em um de dois casos dependendo de qual metade da leitura ocorre a não correspondência.

No primeiro caso, o índice normal é carregado na memória e o mapeador é chamado com a limitação de não realizar substituições na metade direita da leitura. No segundo caso, o índice invertido é carregado na memória e o mapeador é chamado para realizar o mapeamento da mesma leitura, porém invertida, com a limitação de não realizar substituições na metade direita da leitura (metade esquerda da leitura original).

Essas limitações e o uso de dois índices permitem que o retrocesso excessivo seja evitado, ao mesmo tempo em que mantêm alta sensibilidade do processo de mapeamento (LANGMEAD *et al.*, 2009). O número de retrocessos também pode ser limitado pelo parâmetro *--maxbts*, que por padrão vale 125.

1.3.2.2 TOPHAT

TopHat (TRAPNELL; PACHTER; SALZBERG, 2009) é um mapeador de leituras oriundas de RNA-Seq. TopHat identifica junções de *splicing*, o que o torna mais adequado para análises de organismos eucarióticos, uma vez que a maioria dos genes desses organismos apresentam essa característica (IRIMIA *et al.*, 2007).

TopHat tem como base o software Bowtie (LANGMEAD *et al.*, 2009) e, tal como esse, aceita leituras no formato Fasta, Fastq, Csfasta ou Csfastq, além de arquivos de qualidade no formato Qual. TopHat aceita leituras provindas de sequenciamento *single-end* ou *paired-end* , codificadas tanto em espaço de bases como em espaço de cor (por meio do parâmetro *-C/--color*). Como saída TopHat gera um arquivo BAM com os mapeamentos reportados. Esse mapeador também pode fazer uso de múltiplos núcleos de processamento.

A FIGURA 8 mostra as etapas utilizadas pelo TopHat durante sua análise. Primeiramente, o conjunto total de leitura é mapeado contra o genoma de referência utilizando-se o Bowtie. As leituras mapeadas são reportadas como mapeamentos válidos, já as que não mapearem nesse processo são reservadas para análise posterior.

A partir das leituras que de fato mapearam, TopHat monta o consenso das regiões do genoma que foram mapeadas utilizando o módulo de montagem do *software* Maq (LI; RUAN; DURBIN, 2008). Maq também é um *software* mapeador, mas seu foco está em identificar variações nas sequências, como polimorfismos de nucleotídeo único, inserções e deleções (LI; RUAN; DURBIN, 2008).

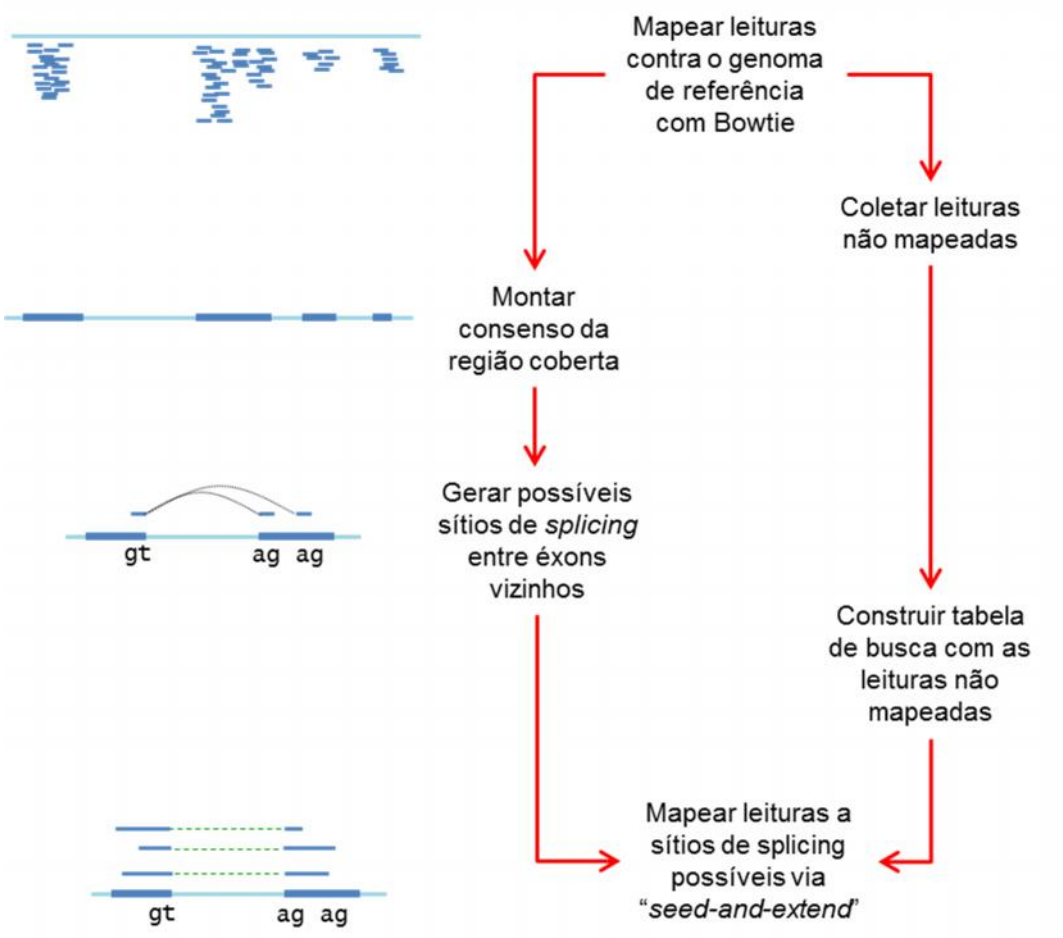


FIGURA 8 – FLUXO DE TRABALHO UTILIZADO POR TOPHAT PARA O MAPEAMENTO E IDENTIFICAÇÃO DE SÍTIOS DE *SPLICING*. FONTE: Modificado de Trapnell, Pachter e Salzberg (2009).

Os consensos montados formam “ilhas” de sequências, os quais constituem prováveis éxons e os espaços localizados entre eles formam prováveis íntrons. Desse modo, TopHat elenca prováveis junções de *splicing* verificando o comprimento dos prováveis íntrons. Essa distância pode ser definida pelo parâmetro *-i/--min-intron-length* (comprimento mínimo de íntron, por padrão é 70) e *-I/--max-intron-length* (comprimento máximo de íntron, por padrão é 500000). TopHat também consideram a presença dos aceptores e doadores canônicos GT-AG presentes entre os prováveis éxons.

Em seguida, para cada provável junção de *splicing*, TopHat realiza uma busca no conjunto de leituras que não mapearam contra o genoma de referência com o objetivo de verificar se alguma leitura transpõe a região de junção entre éxons. TopHat realiza essa tarefa por meio de uma técnica chamada “*seed-and-extend*”,

pela qual as leituras que não mapearam são indexadas em uma tabela de busca. Cada entrada dessa tabela corresponde as possíveis posições dentro da leitura onde essa pode ser particionada de tal maneira que cada parte mapeie em pelo menos *k* bases nas extremidades do par de éxons que compõem a junção. Esse número de bases, *k*, corresponde ao comprimento de uma região chamada de âncora, sendo que esse valor pode ser definido pelo parâmetro *-a/--min-anchor-length* (valor padrão é 8).

Desse modo, TopHat gera uma "semente" concatenando *k* bases da extremidade final do éxon doador com esse mesmo número de bases do início do éxon aceptor. Essa semente é utilizada como chave de busca na tabela das leituras que não mapearam, retornando as posições de partição das leituras. TopHat, então, considerando as posições de partição, mapeia a região de maior qualidade dessas leituras contra a região de âncora dos possíveis éxons, transpondo-as sobre as junções de *splicing* (FIGURA 9). Esse mapeamento também considera o número de não correspondências que podem ocorrer em cada porção da leitura particionada, sendo definido pelo parâmetro *-m/--splice-mismatches* (valor padrão, 0). Por fim, TopHat reporta todos os mapeamentos válidos que ocorreram nas junções de *splicing*.

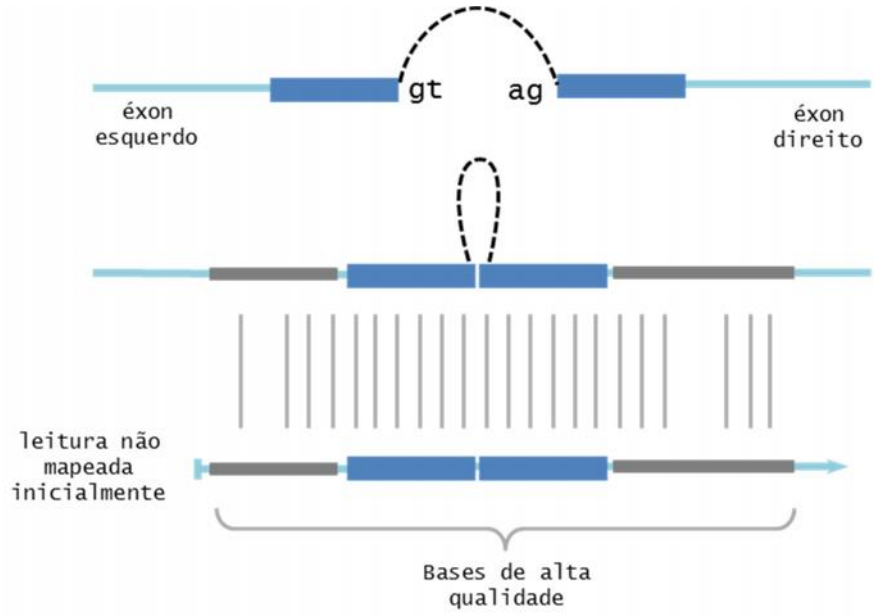


FIGURA 9 – ESTRATÉGIA "*SEED-AND-EXTEND*". Porções azuis escuro representam as regiões de âncora dos possíveis éxons. A concatenação dessas âncoras compõe a "semente" utilizada como chave de busca na tabela de leituras não mapeadas. Porções em cinza encerram a região de alta qualidade da leitura. FONTE: Modificado de Trapnell, Pachter e Salzberg (2009).

1.3.2.3 SAMTOOLS

SAMtools (LI *et al.*, 2009) é uma biblioteca e um *software* para a manipulação de mapeamentos no formato SAM/BAM. Esse software possui funcionalidades de conversão entre formatos de mapeamento, ordenação, indexação, mescla, entre outros. A FIGURA 10 apresenta alguns exemplos de como SAMtools pode ser utilizado. No primeiro exemplo, um mapeamento no formato SAM é convertido para BAM; os parâmetros *-b* e *-S* indicam, respectivamente, que o formato BAM é utilizado como saída e SAM como entrada. No segundo exemplo, o arquivo BAM é ordenado por coordenadas de mapeamento (comportamento padrão); "mapping_sorted" se refere ao sufixo do nome do arquivo que é gerado. No último exemplo, o arquivo BAM, já ordenado, é indexado e é gerado um arquivo chamado "mapping_sorted.bam.bai", o qual contém o esquema de indexação dos mapeamentos.

```
samtools view -b -S mapping.sam > mapping.bam
samtools sort mapping.bam mapping_sorted
samtools index mapping_sorted.bam
```

FIGURA 10 – ALGUMAS OPERAÇÕES QUE PODEM SER REALIZADAS COM SAMTOOLS. Exemplos de conversão para o formato BAM, ordenação e indexação. FONTE: O autor (2013).

1.3.2.4 BEDTOOLS

BEDTools (QUINLAN; HALL, 2010) é um software que possui ferramentas que possibilitam a realização de uma série de operações sobre arquivos de intervalos de características biológicas, como arquivos BAM e GFF. Essas operações são: intersecção, concatenação, contagem, complemento e desordenamento. A FIGURA 11 apresenta um exemplo de utilização do BEDTools

para o cálculo da cobertura de um mapeamento.

BEDTools calcula as coberturas por meio do subcomando *coverage* (FIGURA 11B), para o qual são passados um arquivo de mapeamento no formato BAM (opção *-abam*) e um arquivo com modelos gênico (opção *-b*), como um arquivo no formato GFF. Assim, é calculada a cobertura de um mapeamento sobre a anotação de um genoma.

Como resultado do comando da FIGURA 11B, gera-se um arquivo que contém o mesmo conteúdo do arquivo fornecido em *-b*, acrescido de quatro colunas, representadas pelos campos vermelhos na FIGURA 11C. A primeira coluna contém a cobertura em si, isto é, o número de leituras mapeadas que se sobrepõem ao dado gene. A segunda coluna representa o número de bases que essas leituras cobrem. A terceira coluna equivale ao tamanho do gene em pares de bases. Por fim, a última coluna é a profundidade, ou seja, a proporção do gene que foi coberta (tamanho do gene dividido pelo número de bases).

Vale ressaltar que o cálculo da cobertura é importante para a etapa de sumarização na análise de expressão diferencial, uma vez que a partir desse processo é gerada a tabela com a contagem bruta das leituras que mapearam no genoma de referência, a qual é utilizada como entrada para as ferramentas de análises estatísticas (OSHLACK; ROBINSON; YOUNG, 2010).

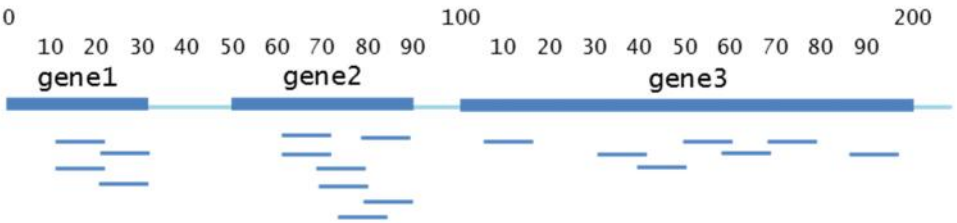
1.3.2.5 DESEQ

DESeq (ANDERS; HUBER, 2010) é uma biblioteca estatística desenvolvida em R (R CORE TEAM, 2012) para análise de expressão diferencial em experimentos de RNA-Seq distribuída pelo projeto Bioconductor (GENTLEMAN *et al.*, 2004).

Essa biblioteca se baseia no princípio de que a contagem de leituras que se encontram associadas a uma classe (genes e transcritos, por exemplo) está linearmente relacionada à abundância dessa classe (MORTAZAVI *et al.*, 2008). DESeq visa realizar testes estatísticos para verificar se o efeito de diferentes

condições biológicas sobre a contagem de leituras ocorreu ou não ao acaso (ANDERS; HUBER, 2010).

A.



B.

```
bedtools coverage -abam mapping.bam -b ref.gff > mapping.cov
```

C.

	Início	Fim	Cobertura	Número de bases	Tamanho do gene	Profundidade
gene1	0	30	4	20	30	0.66666
gene2	50	90	7	30	40	0.75000
gene3	100	200	5	50	100	0.50000

FIGURA 11 – CÁLCULO DA COBERTURA DE MAPEAMENTO UTILIZANDO-SE BEDTOOLS. (A) Mapeamento de leituras em um genoma e em três genes hipotéticos. Os números acima da referência representam coordenadas gênicas. (B) Linha de comando do BEDTools para o cálculo da cobertura de um mapeamento sobre a anotação de um genoma. (C) Em vermelho, os campos calculados por BEDTools, que são, respectivamente, o valor de cobertura, o número de bases que essas leituras cobrem, tamanho do gene em pares de bases e a profundidade. FONTE: O autor (2013).

Para tanto, DESeq modela os dados utilizando-se a Distribuição Binomial Negativa. Essa distribuição oferece um modelo mais realístico para a variação dos dados de contagem em experimentos RNA-Seq em oposição a Distribuição de Poisson (ROBINSON; SMYTH, 2007).

A distribuição de Poisson possui apenas um parâmetro, a média, que define sua variância e todas as suas outras propriedades. O fato da Distribuição de Poisson possuir essa característica faz com que ocorra um fenômeno chamado superdispersão, pelo qual a variabilidade real dos dados é maior do que o modelo de Poisson é capaz de predizer. Além disso, testes estatísticos que se fundamentam nessa distribuição não controlam o erro do tipo I, que é a probabilidade de ocorrerem falsas descobertas (ANDERS; HUBER, 2010; ROBINSON; SMYTH, 2007).

DESeq fundamenta sua modelagem sobre dados de contagem, como

valores inteiros não negativos, os quais são fornecidos por meio de uma tabela, onde as linhas representam as contagens em cada classe analisada e as colunas representam as diferentes condições experimentais (FIGURA 12A). As condições experimentais são sempre duas, controle e tratamento, mas cada condição pode possuir várias réplicas.

A.

	C1	C2	T1	T2	T3
gene1	45	38	14	5	20
gene2	21	24	2	1	2
gene3	839	956	65	19	113
gene4	941	895	96	28	82
gene5	309	327	272	82	406

B.

id	baseMean	baseMeanA	baseMeanB	foldChange	log2 foldChange	pval	padj
gene1	18,947	16,131	20,824	1,291	0,368	0,705	1,000
gene2	5,303	8,648	3,072	0,355	-1,493	0,336	1,000
gene3	196,733	345,014	97,879	0,284	-1,818	0,002	0,045
gene4	210,958	355,265	114,754	0,323	-1,630	0,002	0,036
gene5	282,817	122,580	389,642	3,179	1,668	0,002	0,035

FIGURA 12 – EXEMPLO DE ENTRADA E SAÍDA DO SOFTWARE DE ANÁLISE ESTATÍSTICA DESEQ. (A) Dados de entrada para o DESeq. As colunas representam tratamentos e réplicas, as linhas contêm as contagens brutas das classes para cada tratamento. Notar que há duas réplicas para o controle e três para o tratamento. (B) Dados de saída do DESeq, contendo para cada classe o valor p e *fold change*. FONTE: O autor, 2012.

Como os valores de contagem das classes são diretamente influenciados pela cobertura do sequenciamento, tratamentos com valores de contagem mais elevados podem não representar classes que foram, de fato, diferencialmente expressas, mas uma tendenciosidade introduzida pelo processo de sequenciamento. Para eliminar esse tipo de tendenciosidade, DESeq calcula para todos os tratamentos fatores de tamanho (EQUAÇÃO 2), que são estimativas do tamanho

efetivo dos tratamentos (ANDERS; HUBER, 2010; ROBINSON; SMYTH, 2007).

$$\hat{s}_j = mediana_i \frac{k_{ij}}{\left(\prod_{v=1}^{m} k_{iv}\right)^{1/m}} \quad , \qquad (2)$$

sendo k_{ij} o valor de contagem da classe i referente ao tratamento j, e $(\prod_{v=1}^{m} k_{iv})^{\frac{1}{m}}$ a média geométrica entre os valores de contagem de uma classe i em todos os m tratamentos.

Para calcular a expressão diferencial, DESeq se baseia na estimativa da média e da variância entre condições experimentais. Como o número de réplicas geralmente é limitado, DESeq foi desenvolvido para se ajustar ao desenho experimental do RNA-Seq, que geralmente se compõe de poucas réplicas e muitas classes. Para tanto, ao calcular os valores de média e da variância para cada classe, DESeq escalona os valores de contagem com os fatores de tamanho, ou seja, $k_{ij}/\hat{s}_j$.

Como resultado, DESeq gera uma tabela (FIGURA 12B) contendo os valores de média para as condições experimentais, o *fold change* e seu logaritmo na base 2, o valor p e esse mesmo valor ajustado pelo método de Benjamini-Hochberg. O valor de *fold change* é calculado dividindo-se a média do tratamento pela média do controle para cada classe. O método de Benjamini-Hochberg é um ajuste utilizado em análises que envolvem múltiplos testes de significância, o qual possui o intuito de controlar a taxa de falsas descobertas, ou seja, a proporção de erros causados pela rejeição equivocada da hipótese nula (BENJAMINI; HOCHBERG, 1995).

2 JUSTIFICATIVA

A análise de expressão diferencial em experimentos de *RNA-Seq* possui diversas etapas, as quais possuem diversos *softwares* associados (FIGURA 3). Alguns profissionais, como os da área de ciências biológicas, podem não estar habituados ao uso de tais ferramentas, principalmente pelo fato de que a maioria delas possui interface em linha de comando. Desse modo faz-se necessário uma ferramenta que unifique as etapas da análise de *RNA-Seq*, de tal modo a propiciar uma redução do esforço do usuário para realizar a análise, cabendo a esse praticamente o fornecimento dos arquivos com os dados de entrada, como as leituras e o genoma de referência.

3 OBJETIVOS

3.1 OBJETIVO GERAL

O presente trabalho tem como objetivo desenvolver um pipeline para análise de expressão gênica diferencial em experimentos de *RNA-Seq*.

3.2 OBJETIVOS ESPECÍFICOS

- Unificar em um único software algumas ferramentas utilizadas em bioinformática para análise de *RNA-Seq*;
- Simplificar a análise bioinformática de *RNA-Seq* através do desenvolvimento de um pipeline que contemple diversas ferramentas necessárias para tal;
- Gerar subsídios para uma fácil visualização dos dados obtidos;
- Facilitar a inserção do *pipeline* em outros *workflows*;
- Possibilitar o desenvolvimento de serviços *web* para a análise de *RNA-Seq*.

4 MATERIAIS E MÉTODOS

4.1 AMBIENTE DE DESENVOLVIMENTO

4.1.1 HARDWARE E SISTEMA OPERACIONAL

O computador utilizado para o desenvolvimento do *pipeline* pertence ao Laboratório de Bioinformática do Programa de Pós-graduação em Bioinformática da Universidade Federal do Paraná (UFPR). Esse computador possui um processador Intel® core™ i5 650 (3.20 GHz, dois núcleos de processamento e quatro *threads*), memória RAM de 6 *gigabytes*, um disco rígido de 320 *gigabytes* e sistema operacional Debian GNU/Linux 6.0.6 arquitetura 64 bits.

Também foram utilizados dois *clusteres*: o *cluster* do Laboratório de Bioinformática do Programa de Pós-graduação em Bioinformática da UFPR, intitulado Bioinfo, o qual possui 512 *gigabytes* de memória RAM, 64 processadores; e o *cluster* SGI Altix, intitulado Gauss, pertencente ao Centro Nacional de Supercomputação (CESUP) da Universidade Federal do Rio Grande do Sul, o qual possui 64 unidades de processamento, cada qual com 64 *gigabytes* de memória RAM e dois processadores *dodecacore*, totalizando 1536 núcleos de processamento e 4 *terabytes* de memória RAM.

4.1.2 LINGUAGENS DE PROGRAMAÇÃO

As linguagens de programação utilizadas para o desenvolvimento do pipeline foram as linguagens Python (ROSSUM, 2012) e R (R CORE TEAM, 2012).

4.1.2.1 PYTHON

Python (ROSSUM, 2012) é uma linguagem de programação de alto nível e de propósito geral, a qual enfatiza a produtividade do programador e legibilidade do código. A biblioteca padrão de Python é extensa e possui diversas funcionalidades, como estruturas de dados avançadas (como listas, matrizes e tabelas associativas), manipulação de formatos de arquivos (xml, html, csv, json, por exemplo), empacotamento e compressão de dados, suporte a protocolos de redes (como ftp e http), entre outros. Python foi escolhido como linguagem principal para o desenvolvimento do *pipeline*, uma vez permite que tarefas de gerenciamento do sistema sejam executadas facilmente, como execução e interrupção de processos, verificação de erros. A versão 2.6.6 de Python foi utilizada para o desenvolvimento do *pipeline*.

4.1.2.2 R

R é uma linguagem de programação e um pacote de *software* que possui facilidades para a manipulação e armazenamento de dados, operadores para cálculos sobre arranjos de dados (como matrizes) e facilidades para a geração e exibição de gráficos. O projeto Bioconductor (GENTLEMAN *et al.*, 2004) provê bibliotecas escritas em R para a análise de dados em bioinformática, tal como o DESeq (para análise de expressão diferencial em experimentos de RNA-Seq), o qual foi utilizado no presente trabalho. A versão 2.15.1 de R foi utilizada para o desenvolvimento do *pipeline*.

4.1.3 SOFTWARES

Cinco *softwares* foram utilizados para compor o pipeline: Bowtie (LANGMEAD *et al.*, 2009), TopHat (TRAPNELL; PACHTER; SALZBERG, 2009), SAMtools (LI *et al.*, 2009), BEDTools (QUINLAN; HALL, 2010) e DESeq (ANDERS; HUBER, 2010).

Bowtie é um mapeador de leituras curtas, as quais podem ser oriundas da maioria das plataformas de sequenciamento utilizadas atualmente (FONSECA *et al.*, 2012). A utilização do índice de Burrows-Wheeler para o mapeamento torna a busca rápida e eficiente em termos de uso de memória. De acordo com (FONSECA *et al.*, 2012), Bowtie foi capaz de mapear 798556 leituras de 100 pares de bases de comprimento contra o genoma humano[7] (comprimento de 3,32 gigabases) em, aproximadamente, 169 minutos e utilizando-se de 5 *gigabytes* de memória. Além disso, Bowtie é um *software* utilizado em diversos trabalhos e possui 2077 citações[8] em artigos científicos. Bowtie versão 0.12.7 foi utilizada compor o *pipeline*.

TopHat é um software que utiliza Bowtie como ferramenta de mapeamento e apresenta estratégias de processamento para mapear as leituras em junções de *splicing*, tornando-o um programa mais adequado para se mapear leituras em genomas de referência eucarióticos. TopHat 1.4.1 foi utilizada para compor o *pipeline*.

Pelo fato de Bowtie e TopHat reportarem os mapeamentos no formato SAM e BAM, respectivamente, SAMtools foi escolhido para compor o pipeline por ser a ferramenta de base para manipular esse tipo de formato de arquivo. Além disso, BEDTools também é um *software* que manipula arquivos no formato BAM e apresenta procedimentos automáticos para a contagem de leituras que mapearam em cada característica gênica. Para compor o pipeline foi utilizado SAMtools na versão 0.1.18 e BEDTools versão 2.16.2.

DESeq é uma biblioteca da linguagem de programação R para a análise de expressão gênica diferencial para dados de contagem. O cálculo da significância

[7] http://www.ensembl.org/Homo_sapiens/Info/Index
[8] Dado do Google Acadêmico (scholar.google.com.br) em 21 de fevereiro de 2013.

estatística é baseado na Distribuição Binomial Negativa, a qual se destaca por representar a variabilidade dos dados de modo consistente com a variabilidade real, provendo um controle sobre a proporção de falsas descobertas. DESeq versão 1.8.3 foi utilizada para compor o *pipeline*.

4.2 ESTUDO DE CASO

Leituras oriundas da bactéria *Azospirillum brasilense* fp2 foram utilizadas para que testes fossem realizados sobre o *pipeline*. Assim, duas condições foram testadas para essa bactéria, uma na qual a bactéria apresentava-se em condição de fixação biológica de nitrogênio (tratamento) e outra na qual a bactéria não se encontrava nessa condição (controle), com o objetivo de se verificar qual a diferença de expressão gênica encontrada em comparação a esses estados.

O RNA total das culturas utilizadas no presente trabalho foi extraído, purificado e sequenciado utilizando-se sequenciador SOLiD pertencente ao Departamento de Bioquímica e Biologia Molecular da UFPR. Foram geradas três bibliotecas correspondentes a três réplicas biológicas do tratamento e duas bibliotecas correspondentes a duas réplicas biológicas do controle. Ao total foram geradas 48.424.882 leituras, sendo 27.753.000 para o tratamento e 20.671.882 para o controle, as quais ocuparam um espaço total de 10,5 *gigabytes* no disco rígido (TABELA 6). Todas as leituras geradas possuem o comprimento de 50 pares de bases e estão armazenadas no formato Csfasta e Qual.

O genoma de referência de *A. brasilense* fp2 foi sequenciado pelo Núcleo de Fixação Biológica de Nitrogênio do Departamento de Bioquímica e Biologia Molecular da UFPR e ainda se encontra em processo de deposição no *National Center for Biotechnology Information* (NCBI) [9], portanto ainda não se encontra disponível para o acesso público. A montagem desse genoma de referência compõe-se de 413 *contigs*, totalizando 6.885.108 pares de bases e um percentual de CG de 68% (SILVEIRA, 2012). O arquivo desse genoma no formato Fasta possi

[9] http://www.ncbi.nlm.nih.gov/genome/11059?project_id=182105

6,8 *megabytes*, ao passo que a anotação no formato GFF possui 2,5 *megabytes*.

TABELA 6 – DESCRIÇÃO DAS BIBLIOTECAS DE *Azospirillum brasilense* fp2. Bibliotecas geradas por sequenciamento SOLiD de *A. brasilense* fp2 na condição de fixação biológica de nitrogênio (tratamento) e de não fixação (controle). As bibliotecas controle apresentam duas réplicas, já as de tratamento três réplicas. Os campos de tamanho se referem ao espaço ocupado no disco rígido, em *megabytes*, pelos arquivos de sequência e qualidade das leituras, respectivamente.

BIBLIOTECA	RÉPLICA	NÚMERO DE LEITURAS	TAMANHO (*megabytes*)	
			LEITURAS	QUALIDADES
Controle	1	7318121	486	1096
Controle	2	13353761	886	1999
Tratamento	1	13604507	903	2038
Tratamento	2	3501780	233	523
Tratamento	3	10646713	706	1630
Total		48424882	3214	7286

FONTE: O autor, 2012.

Uma análise dos valores de qualidade das leituras de *A. brasilense* fp2 também foi realizada. Para tanto, utilizou-se um *script* em R (APÊNDICE 1) para se gerar histogramas com a distribuição dos valores de qualidades nas bibliotecas e gráficos *box-plot* do valor de qualidade para cada posição das leituras nas bibliotecas.

O *pipeline* foi instalado nos *clusteres* supracitados para se verificar o tempo de execução do *pipeline* de análise sobre as bibliotecas de *A. brasilense* fp2, tal como o uso de memória. Para tanto, utilizaram-se os programas time 1.7 e top 3.2.8, os quais fazem parte de sistemas operacionais do tipo UNIX, como o Linux, para se realizar as medições no *cluster* Bioinfo. O programa time 1.7 mede o tempo de execução de um processo; já o programa top 3.2.8 é um monitor que fornece informações, como o uso de memória, sobre a execução dos processos que estão sendo executados pelo sistema. Para o *cluster* Gauss, utilizou-se o programa qstat 11.0.2, o qual é um monitor de processos de clusteres que possuem o Torque[10] como gerenciador de recursos e processos. O programa qstat fornece informações sobre tempo de execução e uso de memória.

[10] http://www.adaptivecomputing.com/products/open-source/torque/

5 RESULTADOS E DISCUSSÃO

5.1 O PIPELINE WEDRING

O *pipeline* (FIGURA 13), o qual foi intitulado Wedring, foi implementado como um pacote da linguagem de programação Python e é responsável pelo gerenciamento da execução dos programas envolvidos e de seus códigos de retorno, pelo processamento dos argumentos de linha de comando, pelos arquivos de entrada, saída e de *log* e interceptação de erros (do próprio *pipeline* ou do sistema).

Wedring é composto por quatro etapas: Etapa de Pré-processamento, Etapa de Indexação, Etapa de Mapeamento e Etapa de Expressão Diferencial.

Etapa de Pré-processamento. Corresponde à validação do arquivo de anotação (que deve ser fornecido no formato GFF), com objetivo de se verificar se esse possui o número correto de colunas, de acordo com Durbin e Haussler (2009), e se o caractere de final de linha é compatível com o do sistema. No primeiro caso uma mensagem de erro é emitida e a execução é interrompida. No segundo caso, um novo arquivo é criado substituindo-se o caractere de final de linha por um compatível com o sistema. O nome desse arquivo é o nome do arquivo original acrescido do termo “_validated”. Essa etapa é necessária, uma vez que o software TopHat pode utilizar os modelos gênicos presentes no arquivo GFF para realizar o mapeamento das leituras e BEDTools o utiliza para gerar os arquivos de cobertura. Assim, um arquivo GFF com o formato adequado evita que erros ocorram tardiamente na execução do pipeline. Além disso, no caso de BEDTools, a presença do caractere de final de linha incompatível com o do sistema não acarreta na emissão de um erro, mas na geração de um arquivo de cobertura com formato inadequado, impossibilitando a criação da tabela de contagens.

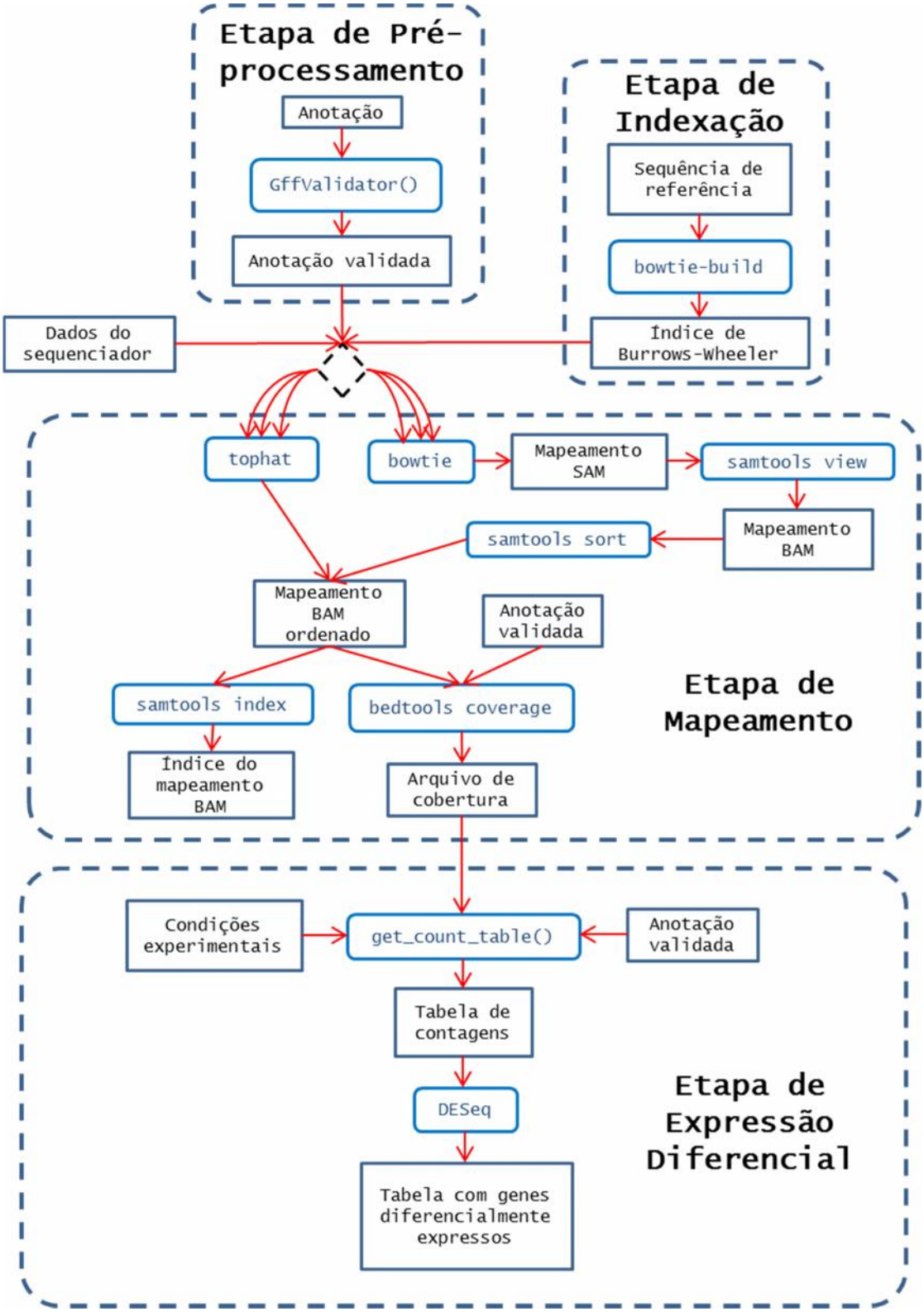


FIGURA 13 – FLUXO DE TRABALHO DO *PIPELINE* WEDRING. O pipeline é composto por quatro etapas (Pré-processamento, Indexação, Mapeamento e Expressão Diferencial), pelas quais uma tabela de genes diferencialmente expressos é gerada a partir de dados de sequenciamento, genoma e anotação de um organismo. Caixas azul-claro indicam programas ou outros algoritmos, já as azul-escuro representam arquivos. Múltiplas setas indicam processamento paralelo. FONTE: O autor, 2012.

Etapa de Indexação. Corresponde à criação do índice de Burrows-Wheeler a partir do genoma de referência, utilizando-se o programa bowtie-build. Essa etapa é opcional uma vez que o usuário pode fornecer diretamente o índice caso esse já fora criado anteriormente.

Etapa de Mapeamento. Corresponde ao mapeamento das leituras fornecidas pelo usuário pelos softwares Bowtie ou TopHat e a manipulação dos mapeamentos gerados. O mapeador a ser utilizado é escolhido pelo usuário, mas de modo geral pode-se dizer que Bowtie (padrão) é mais adequado para organismos procarióticos, ao passo que TopHat o é para os organismos eucarióticos. Essa etapa pode ser executada em processamento paralelo, sendo que cada mapeamento é executado em uma *thread* diferente.

Os mapeamentos gerados por Bowtie são representados no formato SAM, o qual necessita ser convertido para BAM. Essa conversão é realizada por SAMtools utilizando-se o subcomando *view* desse programa. Em seguida, o arquivo BAM gerado necessita ser ordenado, o que é efetivado pelo subcomando *sort* do SAMtools. Essas etapas de conversão e ordenamento não são necessárias para TopHat, já que esse gera mapeamentos em um arquivo BAM ordenado por padrão.

Em seguida, a cobertura do mapeamento é calculada utilizando-se o subcomando *coverage* do programa BEDTools, o qual recebe como entrada um arquivo BAM ordenado e um arquivo de anotação no formato GFF (FIGURA 11). Vale ressaltar que o nome da sequência de referência presente no arquivo de anotação deve ser exatamente igual ao nome presente no arquivo do genoma em si, uma vez que BEDTools utiliza o nome da sequência de referência presente no arquivo de mapeamento para realizar buscas entres as características gênicas presentes no arquivo de anotação. Caso os nomes não coincidam, BEDTools não encontra nenhuma correspondência entre o arquivo de mapeamento e o de anotação, reportando contagens igual a 0.

Ferramentas de visualização de mapeamentos, como o *Integrative Genome Viewer* (IGV) (ROBINSON *et al.*, 2011), necessitam que os arquivos de mapeamento no formato BAM estejam associados a arquivos de índice (THORVALDSDÓTTIR; ROBINSON; MESIROV, 2012). Esse índice é gerado por meio do subcomando *index* do SAMtools, o qual recebe um arquivo BAM ordenado como entrada. Desse

modo, é gerado um arquivo com o mesmo nome do arquivo BAM fornecido, porém acrescido da extensão “bai”. A utilização de índice permite que ferramentas, como o IGV, efetuem acesso randômico aos mapeamentos, ou seja, não necessitam carregar em memória todo o arquivo BAM de uma vez, mas apenas os trechos do arquivo de mapeamento que são requeridos no momento da visualização (THORVALDSDÓTTIR; ROBINSON; MESIROV, 2012).

Etapa de Expressão Diferencial. Ao final da Etapa de Mapeamento, um arquivo de cobertura é gerado para cada mapeamento. Com a informação contida nesses arquivos de cobertura uma tabela contendo as contagens é gerada.

A primeira coluna dessa tabela corresponde a um identificador de cada característica gênica presente no arquivo de anotação. Esse identificador é composto pela palavra “feat” seguido do número da característica gênica. O número da característica gênica é dado pela ordem que essa se encontra no arquivo de anotação. Para que se verifique qual identificador se refere a qual característica gênica, um arquivo com o nome “genomic_features.txt” é gerado contendo essa associação. As colunas subsequentes contêm as condições experimentais fornecidas pelo usuário e as contagens para cada característica gênica. A tabela de contagens é armazenada em um arquivo chamado “count_table.txt”.

O processo de criação da tabela de contagens não é direto, uma vez que o *software* BEDTools não mantém a ordem das características gênicas presentes no arquivo de anotação ao gerar o arquivo de cobertura. Desse modo, Wedring possui algoritmos que identificam à quais características gênicas do arquivo GFF cada linha do arquivo de cobertura corresponde, com base nas informações de característica, início, fim e atributo (colunas 3, 4, 5 e 9, respectivamente, como apresentado na

TABELA 3). A tabela de contagem gerada apresenta as características na mesma ordem em que aparecem no arquivo GFF.

A tabela de contagens é utilizada como entrada para a biblioteca DESeq, a qual gera uma tabela contendo os genes diferencialmente expressos (FIGURA 12B). O usuário também pode solicitar a geração de até três gráficos sobre o cálculo da expressão diferencial: o gráfico de dispersão real e modelada da expressão das características gênicas, *MA-plot* (média de expressão x *fold change*) e *Volcano-plot* (*fold change* x valor p).

Ao final do processamento, Wedring deixa disponível ao usuário os arquivos de mapeamento no formato BAM e seus índices, os arquivos de cobertura, o índice de Burrows-Wheeler (caso esse tenha sido gerado pelo *pipeline*), a tabela de contagem, o arquivo com as características gênicas, os arquivos de log, a tabela reportada por DESeq e as imagens com dados da etapa de expressão diferencial (caso tenha sido solicitas pelo usuário).

Wedring é distribuído sob os termos da licença de software livre GPL v3[11] e seu código fonte pode ser encontrado on-line no repositório GitHub[12].

Com relação aos sistemas operacionais suportados pelo Wedring, não há nenhuma limitação imposta pela implementação do *pipeline*, sendo que basta que o sistema suporte Python para que Wedring possa ser instalado. Contudo, devido aos softwares SAMTools, BEDTools e TopHat não serem instaláveis no sistema operacional Microsoft Windows, o uso do Wedring é possível em sistemas operacionais do tipo UNIX, como o Linux.

5.2 INTERFACE DE LINHA DE COMANDO

As funcionalidades do *pipeline* Wedring estão disponíveis por meio de um programa acessível em linha de comando chamado “wedr”. Esse programa pode receber uma série de parâmetros, os quais estão listados na TABELA 7.

A execução padrão do programa wedr necessita que sejam fornecidos pelo menos os arquivos contendo as leituras, o genoma de referência, a anotação e as condições experimentais;

Os arquivos devem ser fornecidos ao *pipeline* separados por vírgula e sem espaços em branco entre os arquivos. No caso de haver arquivos de qualidade, esses devem ser fornecidos na mesma ordem em que os respectivos arquivos de leitura foram fornecidos.

[11] http://www.gnu.org/licenses/gpl-3.0.html
[12] https://github.com/rcovre/Wedring

TABELA 7 – PARÂMETROS DO PROGRAMA WEDR.

PARÂMETRO	VALOR PADRÃO	DESCRIÇÃO
--help		Mensagem de ajuda.
--version		Versão do *pipeline.*
--quiet		Imprime somente mensagens de erro.
--just-indexbuild		Somente executa a etapa de indexação.
--just-map		Somente executa a etapa de mapeamento.
--just-counttable		Somente gera a tabela de contagens.
--just-de		Somente calcula a expressão diferencial gênica.
-o/--output-dir	./wedr_out	Diretório de saída.
-x/--index-dir	./wedr_index	Diretório onde o índice* será armazenado.
-m/--mapper	bowtie	Nome do *software* mapeador (tophat ou bowtie).
-r/--ref-sequence		Arquivo do genoma de referência.
-i/--bw-index		Nome índice a ser utilizado.
-l/--lib-file		Arquivos de leituras *single-end*.
-1/--pair-mate-1		Arquivos de leituras *paired-end* (par 1).
-2/--pair-mate-2		Arquivos de leituras *paired-end* (par 2).
-q/--quals		Arquivos de qualidade *single-end*.
--q1		Arquivos de qualidade *paired-end* (par 1).
--q2		Arquivos de qualidade *paired-end* (par 2).
-a/--annot-files		Arquivos de anotação.
-g/--coverage-files		Arquivos de cobertura.
-c/--config-file		Arquivo de configuração
-t/--count-table	count_table.txt	Tabela de contagem.
--map-label		Rótulo para o mapeamento.
--index-label		Rótulo para o índice.
-d/--conditions		Condições experimentais.
-n/--num-threads	1	Número de threads.
--group-all		Não considerar arquivos de sequência separadamente.
-p/--path		Caminho a ser adicionado a varial de sistema “PATH”.

FONTE: O autor, 2012.

* O termo índice refere-se ao índice de Burrows-Wheeler.

As condições experimentais são nomes dados para se identificar à quais condições experimentais os arquivos de leituras se referem, portanto ambos devem ser passados na mesma ordem. Quaisquer nomes podem ser utilizados, contanto que sempre sejam de dois tipos, um para se identificar o controle e outro para o tratamento.

Por padrão, o fluxo de trabalho do *pipeline* Wedring é executado completamente, como descrito na FIGURA 13, porém há parâmetros que definem execuções parciais. Com o parâmetro *--just-indexbuild* apenas o índice de Burrows-Wheeler é gerado a partir de um arquivo de genoma de referência, ou seja, apenas

a etapa de indexação é executada. Com o parâmetro *--just-map* somente a etapa de indexação e a etapa de mapeamento são executadas a partir dos arquivos de leituras e suas qualidades, genoma de referência (ou índice de Burrows-Wheeler) e anotação no formato GFF. Com o parâmetro *--just-counttable* é gerada a tabela de contagens a partir de arquivos de cobertura, do arquivo de anotação e dos nomes das condições experimentais. Por fim, o parâmetro *--just-de* executa o programa DESeq a partir da um tabela de contagem e dos nomes das condições experimentais.

O *pipeline* define automaticamente nomes para o índice de Burrows-Wheeler e para os arquivos de mapeamento de acordo com o nome do arquivo de sequência de referência e dos arquivos das bibliotecas, respectivamente. O usuário pode personalizar o nome do índice com o parâmetro *--index-label* e o nome dos mapeamentos com *--map-label*, sendo que o último pode ser fornecido como uma lista de nomes separados por vírgula, tal como os arquivos de leituras.

Quando são fornecidos ao Wedring vários arquivos de leituras e de qualidades (quando for o caso) o pipeline executa um mapeamento para cada arquivo de leitura. Porém, o parâmetro *--group-all* faz com que esses arquivos não sejam considerados separadamente, sendo passados aos programas mapeadores como uma lista de arquivos. Assim, é gerado um único arquivo de mapeamento contendo a informação de todas as leituras utilizadas para o mapeamento. Esse recurso é adequado para quando se deseja utilizar somente a etapa de mapeamento do Wedring (como no uso do parâmetro *--just-map*), pois a informação de qual biblioteca a leitura foi originada é perdida, tornando-se impossível a etapa de expressão diferencial.

Wedring também aceita que seja utilizado um arquivo de configuração para que os parâmetros dos programas associados ao pipeline sejam personalizados. Caso contrário, os softwares são executados com seus parâmetros padrões.

A FIGURA 14 mostra o formato do arquivo de configuração. Nesse formato as seções são marcadas pelo uso dos caracteres colchete (“[” e “]”) e os parâmetros pertencentes a essas são definidos por atribuição utilizando-se o caractere “=”. Linhas iniciadas por cerquilha (“#”) representam comentários, os quais possuem função unicamente informativa, isto é, não afetam a execução do *pipeline*. Contudo,

o único dado que pode ser alterado é o valor do parâmetro, já que alterar o nome da seção e o do parâmetro causa falhas no processamento do arquivo de configuração.

O arquivo de configuração de Wedring possui quatro seções nomeadas de acordo com os programas aos quais os parâmetros se referem. As seções são: bowtie-build, bowtie, tophat e DE.

```
#Comentário

[Seção1]
parâmetro1 = valor
parâmetro2 = valor

[Seção2]
parâmetro3 = valor
```

FIGURA 14 – FORMATO DO ARQUIVO DE CONFIGURAÇÃO UTILIZADO PELO *PIPELINE* WEDRING. FONTE: O autor, 2012.

Os parâmetros das seções bowtie e tophat são processados de acordo como o programa mapeador escolhido pelo usuário, desse modo, constituem seções mutuamente exclusivas. Ou seja, se o usuário optar por utilizar o mapeador TopHat, a seção do Bowtie será ignorada. A seção DE define parâmetros para a etapa de expressão diferencial do *pipeline*, como os do programa DESeq, o nível de significância a ser utilizado, quais imagens serão geradas, entre outros.

Para facilitar o uso, o arquivo de configuração possui comentários explicativos de todos os parâmetros, tal como quais valores podem ser utilizados e a qual parâmetro do programa corresponde. Além disso, os nomes dos parâmetros foram escolhidos para serem o mais parecidos possível com os nomes dos parâmetros dos programas aos quais se referem.

Embora o arquivo de configuração fosse desenvolvido para ser de uso simplificado, há um número muito grande de parâmetros que se referem aos *softwares* mapeadores. Isso se dá devido ao fato desses softwares serem muito flexíveis com relação à configuração das variáveis internas de seus algoritmos, permitido execuções bastante especializadas (FONSECA *et al.*, 2012).

O grande número de parâmetros pode tornar difícil e aumentar a ilegibilidade

do processo de configuração do *pipeline*, principalmente para usuários não habituados com programas que possuem interface em linha de comando. Para resolver esse problema poder-se-ia diminuir o número de opções, mantendo-se apenas as opções mais comumente utilizadas, como número de não correspondências e número de alinhamentos por leitura; tal como a tentativa de se inferir esses parâmetros automaticamente com base no tamanho das leituras, no tipo de organismo que está sendo considerado, por exemplo. Também se poderia utilizar uma interface gráfica para o processo de configuração, pois o usuário não necessitaria entrar em contato direto com o arquivo texto. Além disso, pode haver um esforço para que certos parâmetros relacionados à otimização do processo de mapeamento sejam inferidos automaticamente, de acordo com o tipo de *hardware* em que o *pipeline* encontra-se instalado e aos recursos disponíveis no sistema.

5.3 O PIPELINE COMO UM PACOTE PYTHON

Wedring foi desenvolvido para ser facilmente acoplado em outros *pipelines*. O programa wedr permite que as funcionalidades do pipeline estejam disponíveis para que outros programas possam executá-lo. Além disso, no caso de programas escritos na linguagem de programação Python, o pipeline pode ser acessado como um pacote dessa linguagem.

O *pipeline* Wedring é composto pelos arquivos: a interface de linha de comando, wedr; o arquivo de configuração, wedring.cfg; o script em R que define a etapa de expressão diferencial, diffExprStage.R; e o pacote Python, wedring, que possui a função de gerenciar todas as etapas do *pipeline* (FIGURA 15).

O pacote wedring é composto por oito módulos:

gffutils.py. É utilizado na etapa de pré-processamento e contém a classe *GffValidator*, a qual verifica a validade do arquivo de anotação no formato GFF fornecido;

wedrerror.py. Contém a classe *WedringError*, a qual é a base para todos os erros emitidos pelo sistema;

sysutils.py. Contém funções que são utilizadas em todas as etapas do pipeline e executam tarefas relacionadas ao sistema operacional, como encerramento da execução do *pipeline*, verificação da existência de arquivos no disco rígido, verificação dos programas, entre outros, e também contém a classe *BioSoft*, a qual define a interface entre o *pipeline* e os programas que o compõe;

indexstage.py. Define a classe etapa de indexação por meio da classe *IndexBuilder*, responsável pelo processamento dos parâmetros e execução do programa bowtie-build;

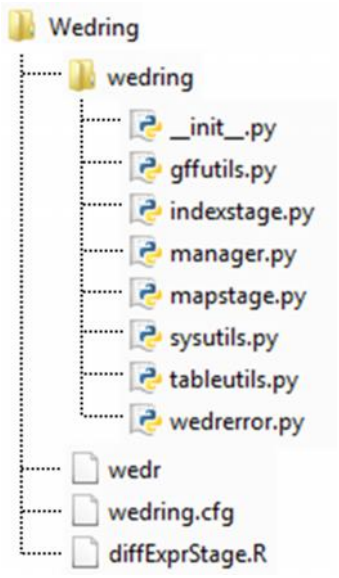


FIGURA 15 – ARQUIVOS QUE COMPÕE O *PIPELINE* WEDRING. O *pipeline* Wedring é composto pelos arquivos wedr, wedring.cfg, diffExprStage.R e o pacote da linguagem de programação Python wedring. FONTE: O autor, 2012.

mapstage.py. Define etapa de mapeamento por meio da classe *WedringMast*, a qual é responsável pelo processamento dos parâmetros e execução dos *softwares* Bowtie, Tophat, SAMtools e BEDTools;

tableutils.py. Possui funções para a formatação e escrita em disco da tabela de contagem;

manager.py. Gerencia todas as etapas do *pipeline*, isto é, o processamento dos parâmetros fornecidos em linha de comando, inicialização e execução das etapas e coordenação entre entradas e saídas dos programas, os quais são definidos pela classe *Wedring*;

__init__.py. Corresponde a um módulo necessário para se definir pacotes em Python (ROSSUM, 2012), podendo conter informações para inicializar o pacote. No caso do Wedring, esse módulo importa o conteúdo de todos os outros módulos que compõe o pipeline para que esses estejam disponíveis diretamente no espaço de nomes do pacote wedring.

O arquivo diffExprStage.R é um *script* escrito em R que contém chamadas as funções da biblioteca DESeq. As funções do DESeq que são executadas são: *newCountDataSet*, que gera uma estrutura inicial com os dados da tabela de contagens e das condições experimentais; *estimateSizeFactors*, a qual calcula o fatores de tamanho, que são utilizados como elementos de normalização; *estimateDispersions*, a qual calcula a dispersão dos dados; e *nbinomTest*, a qual calcula a expressão diferencial dos dados a partir dos valores de dispersão.

A utilização do *pipeline* Wedring dentro de outros programas escritos em Python se dá inicialmente pela sua instalação no sistema. Para tanto, há um *script* de instalação que é distribuído conjuntamente ao *pipeline*, chamado setup.py. O processo de instalação é realizado pela chamada desse script em linha de comando fornecendo-lhe o subcomando *install* (FIGURA 16A). Para que a instalação ocorra com sucesso é necessário ao menos que Python 2.6 esteja instalado na máquina.

Para a utilização do *pipeline* é necessário que estejam instalados no sistema os programas Bowtie e/ou Tophat, SAMtools, BEDTools e DESeq (a partir da versão 1.5.1 e, por consequência, R 2.14 em diante deve estar instalado também). O uso (FIGURA 16B) inicialmente se dá pela importação do pacote wedring, o qual disponibiliza as funções e classes do pipeline para o interpretador Python. Em seguida, é necessária a instanciação da classe *Wedring*, a qual é descrita em manager.py e é responsável pela unificação e gerenciamento de todas as etapas do *pipeline*. A classe *Wedring* é instanciada utilizando-se como argumentos os parâmetros descritos na TABELA 7 e os dados a serem processados. A execução do *pipeline* é, então, iniciada por meio da chamada ao método *run* da classe *Wedring*.

```
A.
    python setup.py install


B.
    import wedring

    ref = 'ref.fasta'    # genoma de referência
    annot = 'ref.gff'    # arquivo de anotação

    # leituras e arquivos de qualidade
    libs = 'lib_C1.csfasta,lib_C2.csfasta,lib_T1.csfasta,lib_T2.csfasta'
    quals = 'lib_C1.qual,lib_C2.qual,lib_T1.qual,lib_T2.qual'

    conds = 'C,C,T,T'    # condições experimentais

    config = 'wedring.cfg'    # arquivo de configuração

    args = ['-r', reference, '-a', annotation, '-l', libs, '-q', quals,
            '-d', conditions, '-c', config]

    wedr = wedring.Wedring(args)
    wedr.run()
```

FIGURA 16 – INSTALAÇÃO E USO DO PIPELINE WEDRING EM PROGRAMAS PYTHON. (A) A instalação do pipeline é feita executando-se o *script* setup.py com o subcomando install. (B) Uso de Wedring exemplificado com dados fictícios, considerando-se duas bibliotecas de controle e duas de tratamento oriundas de sequenciamento SOLiD. FONTE: O autor, 2012.

5.4 ESTUDO DE CASO

Wedring foi utilizado para se analisar os dados de expressão gênica de *A. brasilense* fp2, adquiridos por meio de experimentos de RNA-Seq sequenciados na plataforma SOLiD. O *pipeline* foi executado no *cluster* Bioinfo. Como *A. brasilense* fp2 é uma bactéria, o *software* Bowtie (padrão para o Wedring) foi utilizado como mapeador. As leituras oriundas da plataforma SOLiD são codificadas no espaço de cor e arquivos de qualidade também são fornecidos.

Desse modo, o parâmetro *-C* necessitou ser fornecido para os programas Bowtie e bowtie-build para indicar o uso de sequências em espaço de cor, e --integer-quals para indicar que os valores de qualidade são números inteiros. O modo de mapeamento escolhido foi o modo "n", para que os valores de qualidade fossem considerados, permitindo-se o máximo de três não correspondências na região de maior qualidade das leituras, utilizando-se o parâmetro *-n 3*. Foi permitido que cada

leitura mapeasse em apenas um local no genoma de referência, utilizando-se o parâmetro *-m 1*. Os parâmetro *--al*, *--un* e *--max* foram fornecidos para que Bowtie reportasse, além dos mapeamentos, quais leituras mapearam, não mapearem ou foram suprimidas, respectivamente, gerando arquivos de leituras e de valores de qualidade para cada um desses casos. Vale ressaltar que leituras suprimidas são aquelas às quais os mapeamentos não são reportados por apresentarem um número de mapeamentos que excede o valor definido pelo parâmetro *-m*. Além disso, indicou-se por meio do parâmetro *-p 6* que Bowtie pôde utilizar-se de seis *threads* para cada mapeamento. Também foi solicitado ao *pipeline* que fosse gerado o gráfico *Volcano-plot* com os dados de expressão gênica. Os parâmetros supracitados são fornecidos aos programas a que se referem por meio do arquivo de configuração (FIGURA 17A).

Assim, o programa wedr foi executado (FIGURA 17B) com o genoma de referência (*-r*), anotação (*-a*), leituras (*-l*) e arquivos de qualidade (*-q*) de *A. brasilense* fp2. Além disso, foram fornecidas as condições experimentais (*-d*) na mesma ordem em que as leituras a que se referem aparecem, isto é, dois controles seguidos de três tratamentos. O arquivo de configuração foi passado pelo parâmetro *-c*. E, por fim, o parâmetro *-n* indica que cinco *threads* foram utilizadas para se realizar a etapa de mapeamento, ou seja, cinco *threads* são iniciadas e cada uma dessas utilizam outras seis *threads* (dado fornecido pelo arquivo de configuração) para se realizar os cinco mapeamento que compõe a análise, totalizando um uso de 30 *threads*.

Ao final do processo de análise do *pipeline* foram reportados 95 genes diferencialmente expressos, sendo 51 sobreexpressos e 44 subexpressos (FIGURA 18). A análise nos valores de qualidade das leituras foi realizada, uma vez que o arquivo de anotação de *A. brasilense* fp2 possui 6211 características gênicas, muito mais do que o número de genes diferencialmente expressos reportados. No presente trabalho são figurados apenas os dados acerca da biblioteca Controle-1 de *A. brasilense* fp2 por questão de espaço e pelo fato das outras bibliotecas terem seguido padrão semelhante em seus valores de qualidade.

A.

```
[bowtie-build]
color = true

[bowtie]
query_in_file = fasta
color = true
quals = true
integer_quals = true
n = 3
m_ = 1
al = true
un = true
max = true
threads = 6

[DE]
volcano_plot = true
```

B.

```
wedr -n 5
     -r abfp2.fasta
     -a abfp2.gff
     -l abfp2_C1.csfasta,abfp2_C2.csfasta,abfp2_T1.csfasta,¶
        abfp2_T2.csfasta,abfp2_T3.csfasta
     -q abfp2_C1.qual,abfp2_C2.qual,abfp2_T1.qual,abfp2_T2.qual,¶
        abfp2_T3.qual
     -d C,C,T,T,T
     -c wedring.cfg
```

FIGURA 17 – LINHA DE COMANDO DO WEDRING E PARÂMETROS UTILIZADOS NA ANÁLISE DE *A. brasilense* fp2. (A) Parâmetros utilizados no arquivo de configuração. (B) Linha de comando do pipeline; para ilustração cada linha representa um parâmetro e o caractere ¶ para uma linha que precisou ser quebrada, contudo, em uma situação real, os parâmetros devem ser passados em uma única linha. FONTE: O autor, 2012.

Os valores de qualidade das leituras de *A. brasilense* fp2 possuem uma grande quantidade de valores de baixa qualidade, evidenciado principalmente pelas leituras que não mapearam (FIGURA 19C). Vale ressaltar que os valores de qualidade de maior valor estão em maior quantidade para as leituras mapeadas (FIGURA 19A). Por outro lado, uma menor quantidade de valores de alta qualidade, como para as leituras suprimidas (FIGURA 19B), aumenta a chance de se ocorreram mapeamentos que ocorram ao acaso, daí a importância de se limitar o número de mapeamentos permitidos por leitura e de se limitar número e valor de qualidade para as não correspondências.

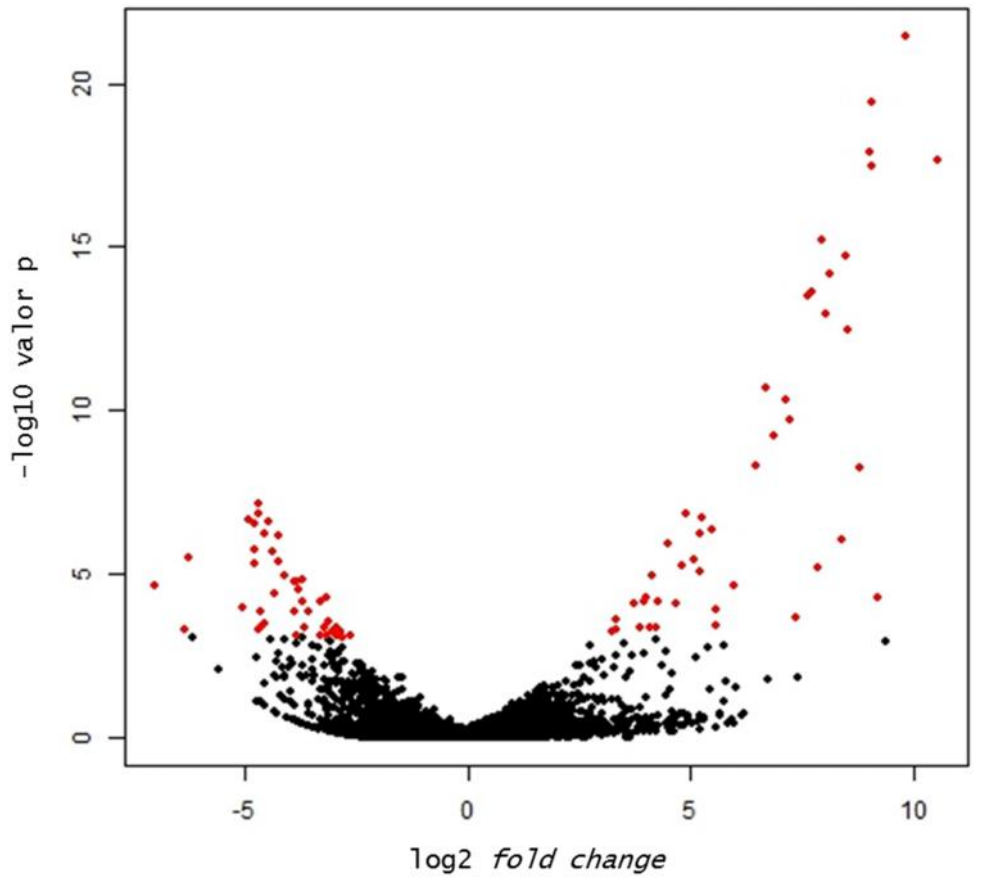


FIGURA 18 – *VOLCANO-PLOT* DOS DADOS DE *A. brasilense* fp2. Pontos vermelhos indicam genes diferencialmente expressos, sendo genes sobreexpressos aqueles com valor de *fold change* maior que 0 e subexpresso aqueles com valor menor que 0. FONTE: O autor, 2012.

Quando se analisa os valores de qualidade por posição das leituras (FIGURA 20), as leituras que mapearam (FIGURA 20A) são as que apresentam valores de qualidade maiores, além de uma menor variação nesse valor, ao passo que as leituras que foram suprimidas e as não mapeadas (FIGURA 20B e FIGURA 20C, respectivamente) possuíram variações maiores em todas as posições das leituras.

Desse modo, pela observação da FIGURA 20A, os valores de qualidade se mantêm elevados e aproximadamente constantes até a base 30. Assim um novo ciclo de análise foi realizado, mas agora se fornecendo o parâmetro --trim3 para o mapeador Bowtie com o valor de 20 (a linha *trim3 = 20* foi adicionada ao arquivo de configuração).

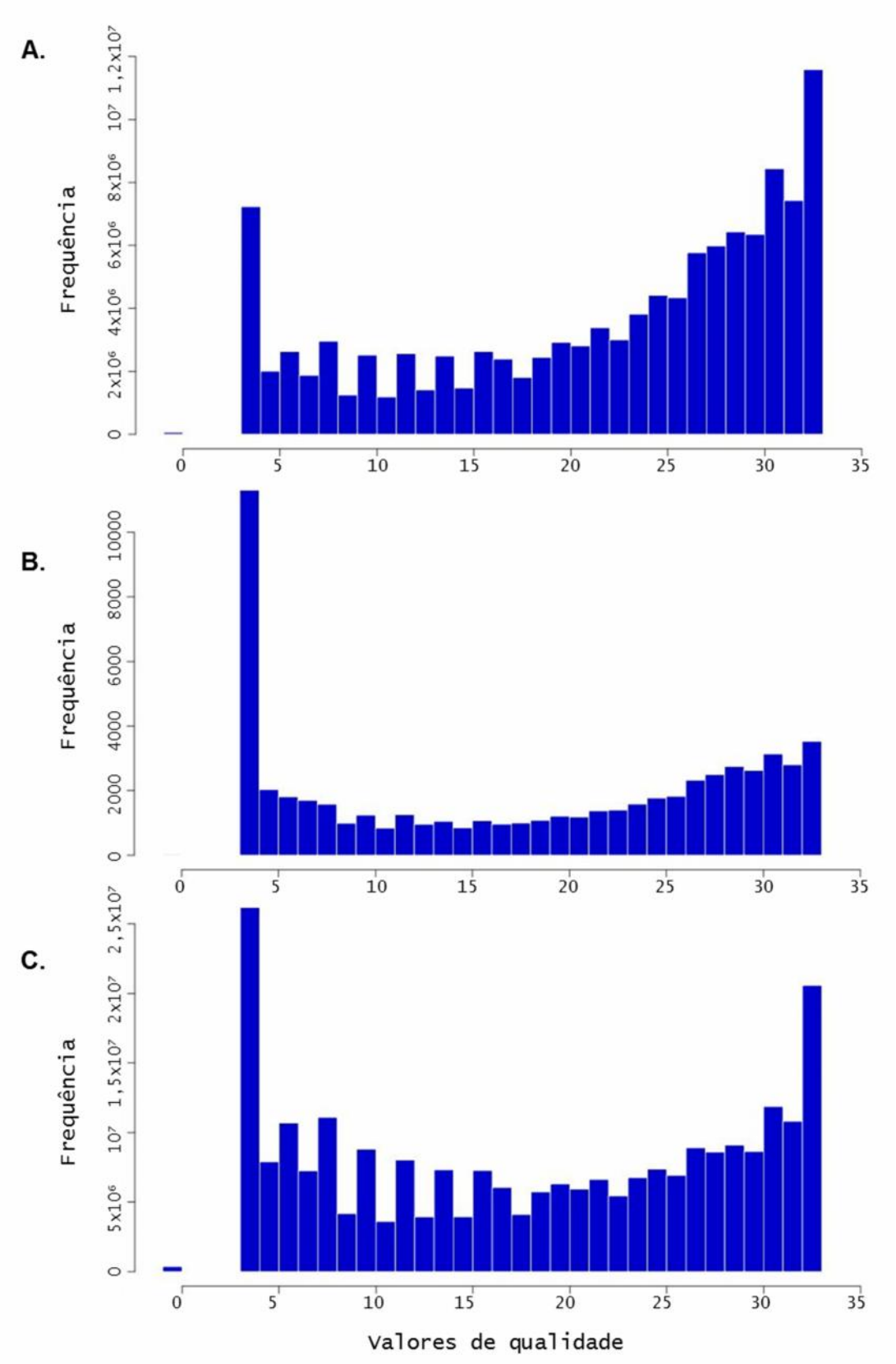


FIGURA 19 – DISTRIBUIÇÃO DOS VALORES DE QUALIDADE DAS LEITURAS DA BIBLIOTECA CONTROLE-1 DE *A. brasilense* fp2. Distribuição dos valores de qualidade para as leituras mapeadas (A), para as leituras suprimidas (B) e para as leituras que não mapearam (C). FONTE: O autor, 2012.

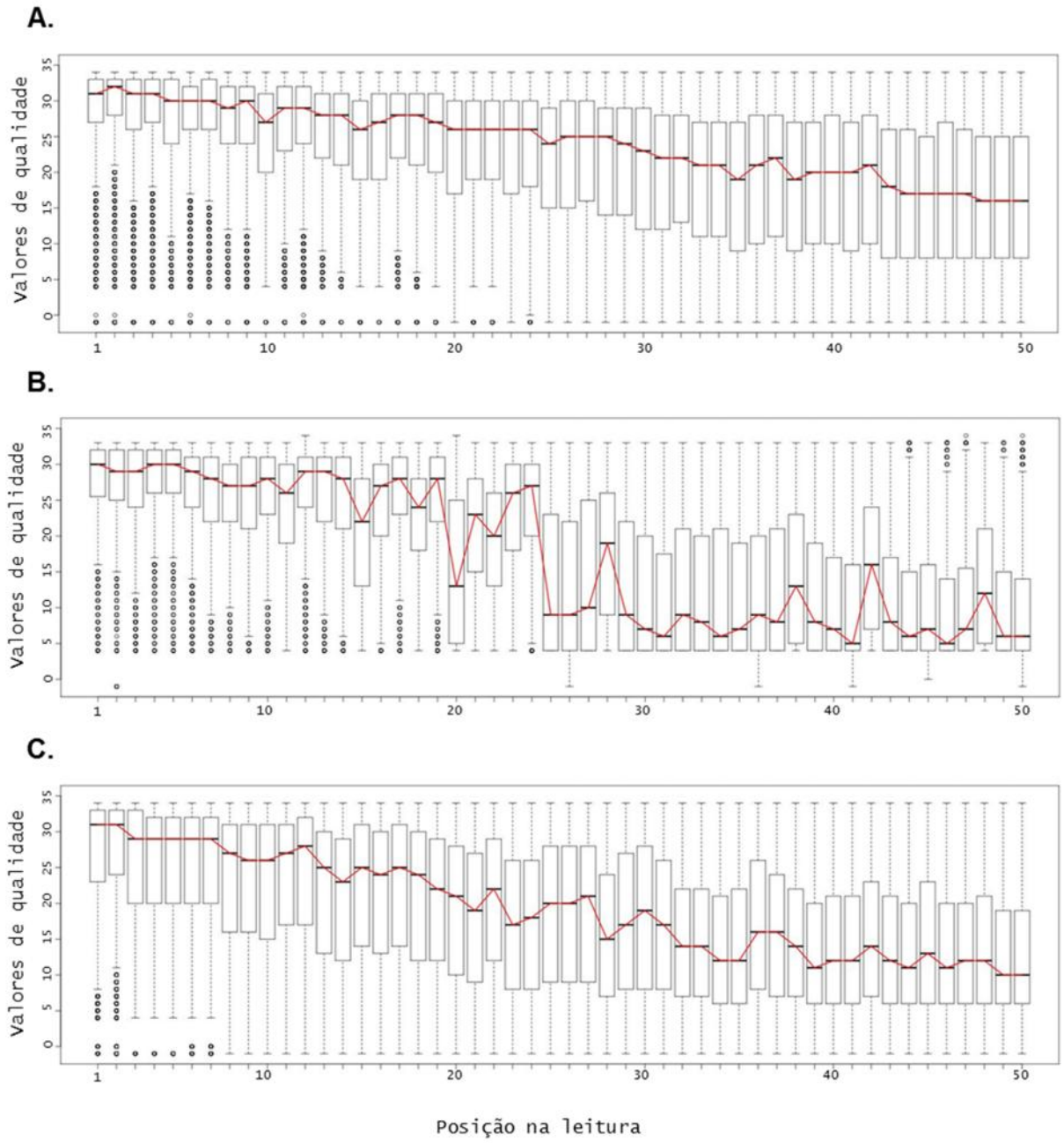


FIGURA 20 – BOX-PLOT DOS VALORES DE QUALIDADE POR POSIÇÃO NAS LEITURAS DA BIBLIOTECA CONTROLE-1 DE *A. brasilense* fp2. Valores de qualidade por posição nas leituras mapeadas (A), nas leituras suprimidas (B) e nas leituras que não mapearam (C). A linha vermelha passa pelo valor de mediana da qualidade para cada posição das leituras. Círculos pretos indicam *outliers*. FONTE: O autor, 2012.

O uso do parâmetro *--trim3* faz com que sejam removidos um dado número de bases da leitura a partir da extremidade 3' antes que o processo de mapeamento seja realizado.

Ao final do processo de análise do *pipeline* foram reportados 141 genes

diferencialmente expressos, sendo 67 sobreexpressos e 74 subexpressos (FIGURA 21).

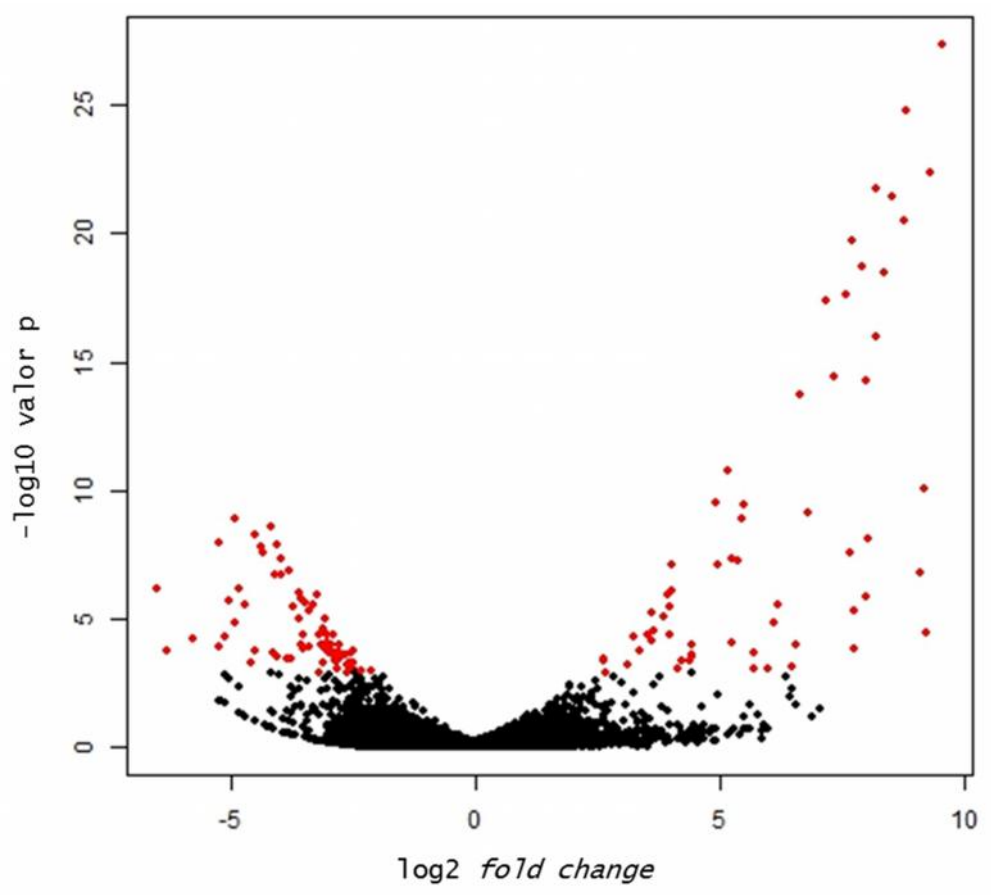


FIGURA 21 - *VOLCANO-PLOT* DOS DADOS DE *A. brasilense* fp2 APÓS REMOÇÃO DE 20 BASES DA EXTREMIDADE 3' DAS LEITURAS. . Pontos vermelhos indicam genes diferencialmente expressos, sendo genes sobreexpressos aqueles com valor de *fold change* maior que 0 e subexpresso aqueles com valor menor que 0. FONTE: O autor, 2012.

O uso do parâmetro *--trim3* gerou um aumento no número de leituras que mapearam tal como nas que foram suprimidas e houve uma diminuição no número de leituras que falharam em mapear (FIGURA 22).

A remoção de bases da extremidade 3' da leitura exclui as bases de baixa qualidade da leitura, ou seja, bases que possuem uma probabilidade maior de terem sido sequenciadas erroneamente, com o objetivo de diminuir o número de mapeamentos falso-positivos (LEE *et al.*, 2012).

Esse fato é evidenciado no segundo ciclo de análise pelo aumento no número de leituras mapeadas (FIGURA 22A) juntamente com o aumento na frequência de valores de qualidade mais elevados, tanto no conjunto geral de

leituras mapeadas (FIGURA 23A) como na extremidade 5’ dessas (FIGURA 24A).

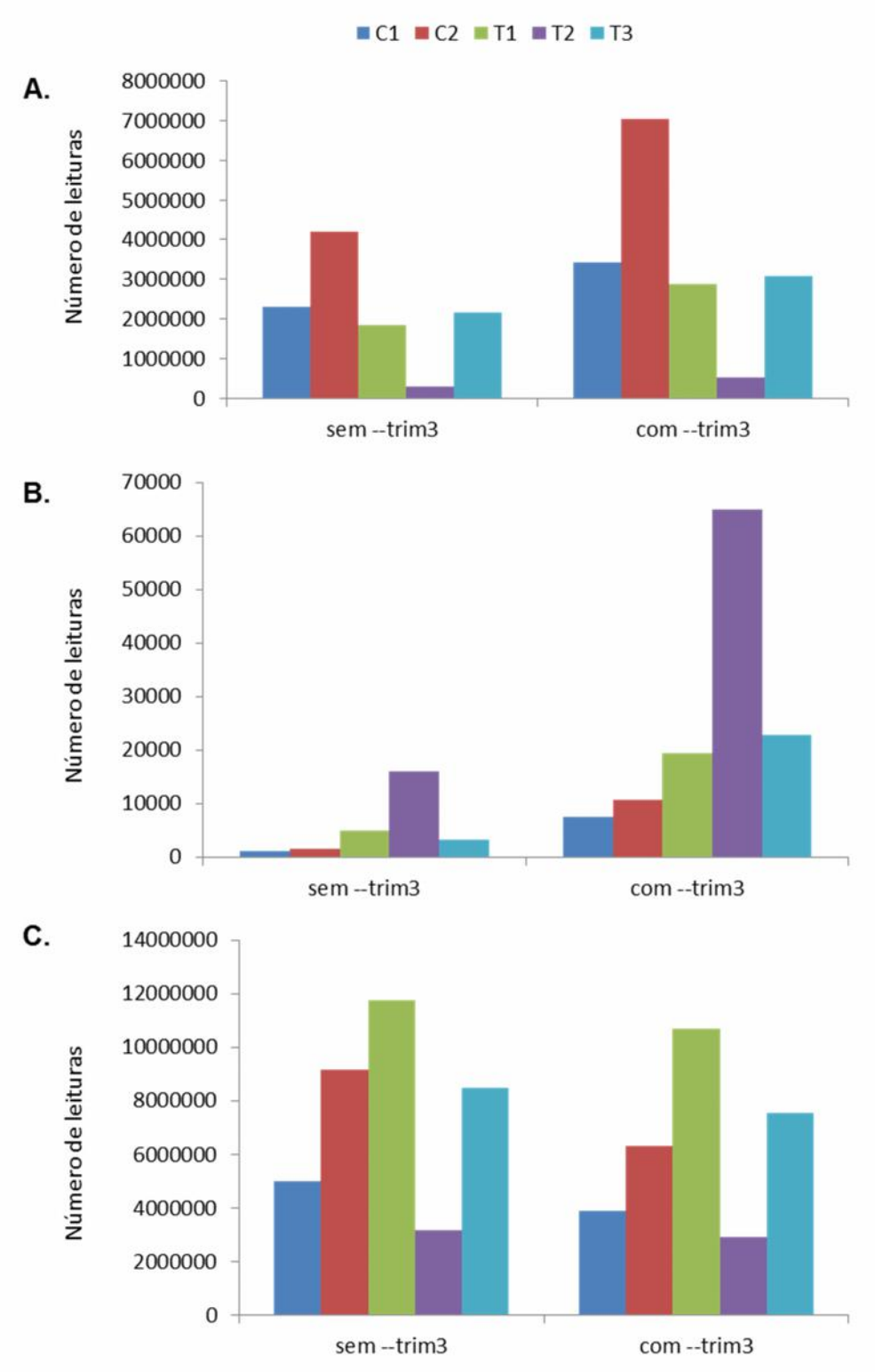


FIGURA 22 – COMPARAÇÃO DO NÚMERO DE LEITURAS ENTRE MAPEAMENTOS COM E SEM O PARÂMETRO *--trim3*. As quantidades referem-se às leituras mapeadas (A), suprimidas (B) e não mapeadas (C). C1, C2, T1, T2, T3 referem-se às condições experimentais das bibliotecas de *A. brasilense* fp2, a dizer, dois controles e três tratamentos. FONTE: O autor, 2012.

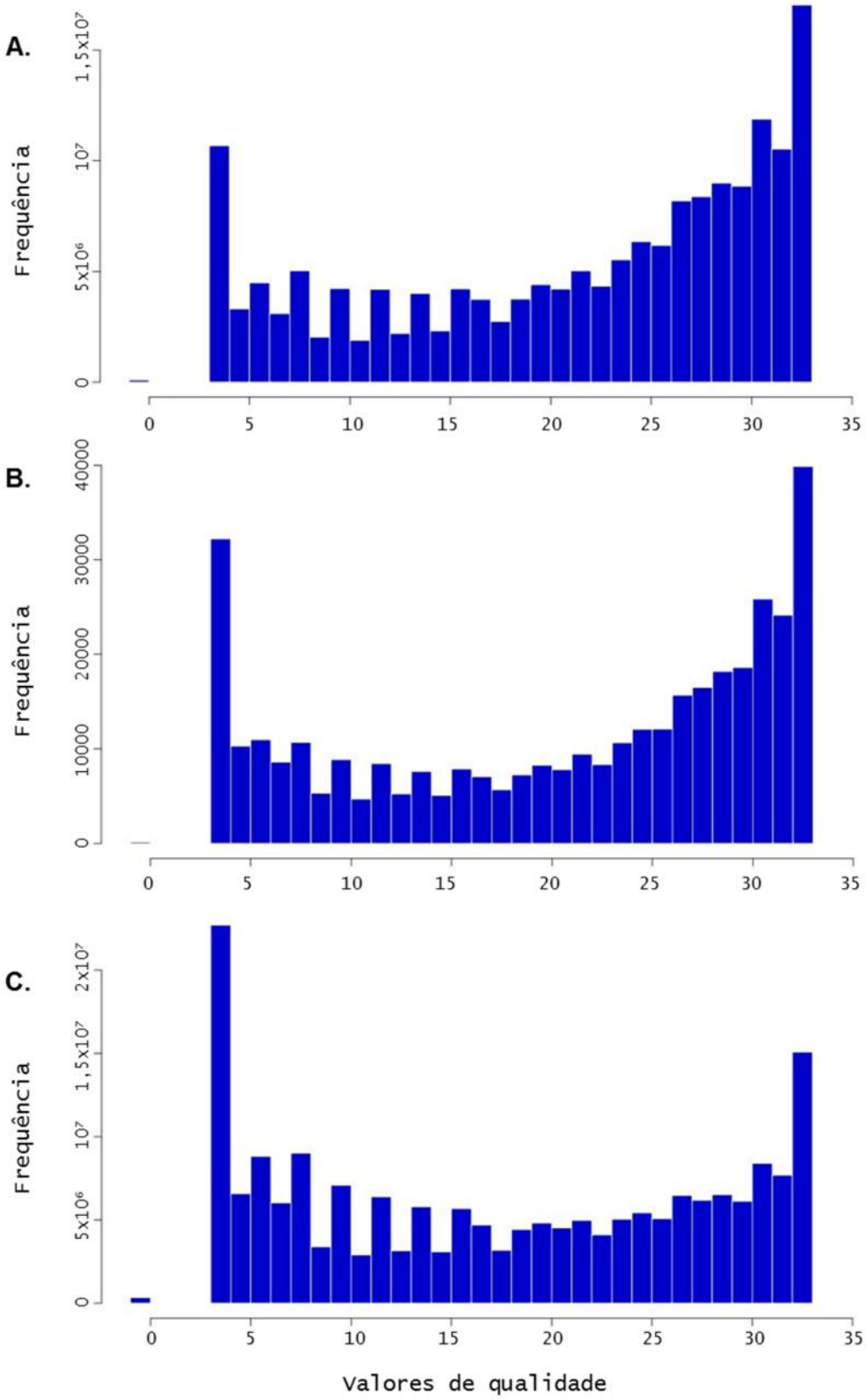


FIGURA 23 – DISTRIBUIÇÃO DOS VALORES DE QUALIDADE DAS LEITURAS DA BIBLIOTECA CONTROLE-1 DE *A. brasilense* fp2 APÓS REMOÇÃO DE 20 BASES DA EXTREMIDADE 3' DAS LEITURAS. Distribuição dos valores de qualidade para as leituras mapeadas (A), para as leituras suprimidas (B) e para as leituras que não mapearam (C). FONTE: O autor, 2012.

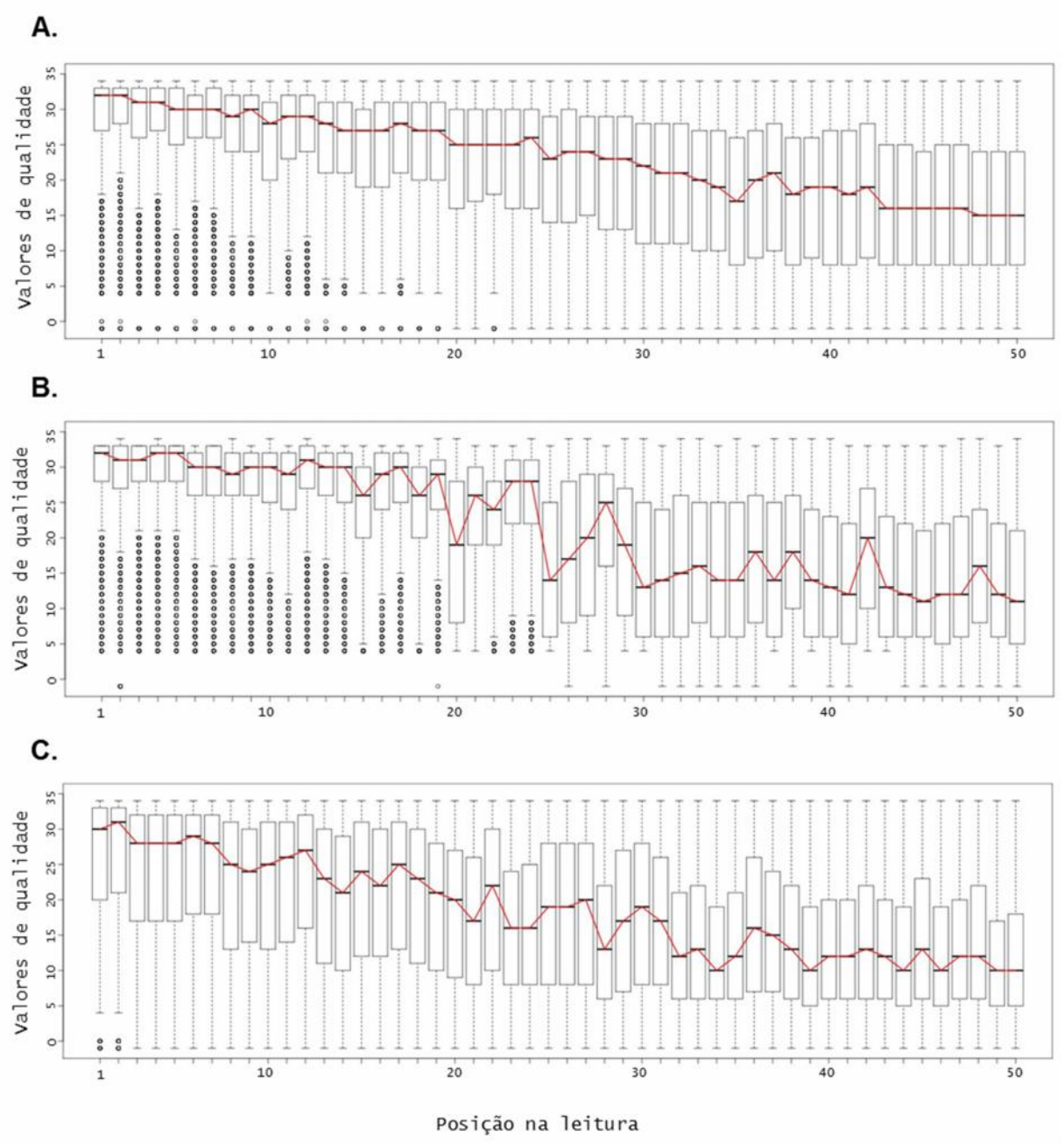


FIGURA 24 – BOX-PLOT DOS VALORES DE QUALIDADE POR POSIÇÃO NAS LEITURAS DA BIBLIOTECA CONTROLE-1 DE *A. brasilense* fp2 APÓS REMOÇÃO DE 20 BASES DA EXTREMIDADE 3' DAS LEITURAS. Valores de qualidade por posição nas leituras mapeadas (A), nas leituras suprimidas (B) e nas leituras que não mapearam (C). A linha vermelha passa pelo valor de mediana da qualidade para cada posição das leituras. Círculos pretos indicam *outliers*. FONTE: O autor, 2012.

Além disso, a diminuição do número de valores de qualidade *outliers* (FIGURA 24A) nas leituras que foram mapeadas, associado ao aumento na frequência de qualidades mais altas nas leituras suprimidas (FIGURA 23B), indica que Bowtie foi mais seletivo em relação às leituras que foram mapeadas. Desse

modo, esses fatos geram indícios de que foi baixa a probabilidade de ocorrência de mapeamentos ao acaso devido ao tamanho reduzido das leituras (30 pares de based após a remoção de 20 pares).

As leituras suprimidas (FIGURA 24B) e as não mapeadas (FIGURA 24C) possuem uma variação muito alta em seus valores de qualidade, além de muitos *outliers*, o que é um indicativo de que as leituras de pior qualidade foram, de fato, excluídas do mapeamento.

Portanto, o uso do parâmetro *--trim3* possibilitou que um número maior de leituras com qualidade mais alta fossem mapeadas, aumentando o número de genes diferencialmente expressos reportados e também a probabilidade de que esses genes representem as condições fisiológicas reais de *A. brasilense* fp2.

Assim, o *pipeline* Wedring possibilitou em uma única linha de comando que todas as etapas descritas na FIGURA 13 fossem realizadas. Caso Wedring não fosse utilizado, muitos comandos necessitariam ser executados em sequência para se realizar o mesmo fluxo de análise do *pipeline* (FIGURA 25), o que dificultaria a análise de expressão gênica em RNA-Seq.

Ademais, os dados gerados pelo *pipeline* permitem análises posteriores com outras ferramentas, como a análise de qualidades, e aquelas que possam ser realizadas sobre os dados dos genes diferencialmente expressos, como a de ontologias e a de vias metabólicas.

5.4 ANÁLISE DE DESEMPENHO

A análise tempo de execução e uso de memória foi executada com os mesmos parâmetros utilizados para se realizar o segundo ciclo de análise de *A. brasilense* fp2, porém utilizando apenas uma *thread*. Os resultados são apresentados na TABELA 8.

```
$ bowtie-build -C abfp2.fasta abfp2

$ bowtie -C -m 1 -n 3 -f -3 20 -S -Q abfp2_C1.qual abfp2 abfp2_C1.csfasta abfp2_C1_vs_abfp2.sam
$ bowtie -C -m 1 -n 3 -f -3 20 -S -Q abfp2_C2.qual abfp2 abfp2_C2.csfasta abfp2_C2_vs_abfp2.sam
$ bowtie -C -m 1 -n 3 -f -3 20 -S -Q abfp2_T1.qual abfp2 abfp2_T1.csfasta abfp2_T1_vs_abfp2.sam
$ bowtie -C -m 1 -n 3 -f -3 20 -S -Q abfp2_T2.qual abfp2 abfp2_T2.csfasta abfp2_T2_vs_abfp2.sam
$ bowtie -C -m 1 -n 3 -f -3 20 -S -Q abfp2_T3.qual abfp2 abfp2_T3.csfasta abfp2_T3_vs_abfp2.sam

$ samtools view -bS -o abfp2_C1_vs_abfp2.bam abfp2_C1_vs_abfp2.sam
$ samtools view -bS -o abfp2_C2_vs_abfp2.bam abfp2_C2_vs_abfp2.sam
$ samtools view -bS -o abfp2_T1_vs_abfp2.bam abfp2_T1_vs_abfp2.sam
$ samtools view -bS -o abfp2_T2_vs_abfp2.bam abfp2_T2_vs_abfp2.sam
$ samtools view -bS -o abfp2_T3_vs_abfp2.bam abfp2_T3_vs_abfp2.sam

$ samtools sort abfp2_C1_vs_abfp2.bam abfp2_C1_vs_abfp2
$ samtools sort abfp2_C2_vs_abfp2.bam abfp2_C2_vs_abfp2
$ samtools sort abfp2_T1_vs_abfp2.bam abfp2_T1_vs_abfp2
$ samtools sort abfp2_T2_vs_abfp2.bam abfp2_T2_vs_abfp2
$ samtools sort abfp2_T3_vs_abfp2.bam abfp2_T3_vs_abfp2

$ samtools index abfp2_C1_vs_abfp2.bam
$ samtools index abfp2_C2_vs_abfp2.bam
$ samtools index abfp2_T1_vs_abfp2.bam
$ samtools index abfp2_T2_vs_abfp2.bam
$ samtools index abfp2_T3_vs_abfp2.bam

$ bedtools coverage -s -abam abfp2_C1_vs_abfp2.bam -b abfp2.gff > abfp2_C1_vs_abfp2.cov
$ bedtools coverage -s -abam abfp2_C2_vs_abfp2.bam -b abfp2.gff > abfp2_C2_vs_abfp2.cov
$ bedtools coverage -s -abam abfp2_T1_vs_abfp2.bam -b abfp2.gff > abfp2_T1_vs_abfp2.cov
$ bedtools coverage -s -abam abfp2_T2_vs_abfp2.bam -b abfp2.gff > abfp2_T2_vs_abfp2.cov
$ bedtools coverage -s -abam abfp2_T3_vs_abfp2.bam -b abfp2.gff > abfp2_T3_vs_abfp2.cov

$ # Ferramenta para a criação da tabela de contagens "count_table.txt"

$ R
> library(DESeq)
> cnt.table <- read.table('count_table.txt', header=TRUE)
> cds <- newCountDataSet(cnt.table, c('C', 'T'))
> cds <- estimateSizeFactors(cds)
> cds <- estimateDispersions(cds)
> res <- nbinomTest(cds, 'C', 'T')
> write.table(res, 'diffexpr.txt', sep='\t', quote=FALSE, row.names=FALSE)
```

FIGURA 25 – COMANDOS NECESSÁRIOS PARA A ANÁLISE DOS DADOS DE RNA-SEQ DE *A. brasilense* fp2 SEM A UTILIZAÇAO DO *PIPELINE* WEDRING. Linha iniciadas com o caractere “$” indicam comando executados em terminal, ao passo que as iniciadas com “>” são executadas dentro do ambiente R. Os parâmetros *-p*, *--integer-quals*, *--al*, *--un* e *--max* foram omitidos das linhas de comando do Bowtie por questão de espaço. A criação da tabela de contagens não é direta, como descrito no presente trabalho, portanto não é possível apresentar um exemplo trivial de como gerá-la. No caso do Wedring, é utilizado o módulo ‘tableutils.py’. FONTE: O autor, 2012.

Embora tenham sido processados os mesmos dados em ambos os *clusteres*, as diferenças encontradas entre os tempos de execução e uso de memória podem ter sido consequência de: (1) o fato dos *clusteres* possuírem recursos computacionais distintos, tal como o sistema gerencia esses recursos; (2) terem sido usadas ferramentas distintas para o aferimento das medidas (time e top para o *cluster* Bioinfo e qstat para o Gauss); e (3) não haver acesso exclusivo aos *clusteres*, uma vez que esses estão disponíveis para que outros usuários os utilizem, sendo

que pode ter havido outros processos não relacionados ao Wedring sendo executados concomitantemente ao *pipeline*.

TABELA 8 – TEMPO DE EXECUÇÃO E USO DE MEMÓRIA PELO *PIPELINE* WEDRING NOS *CLUSTERES* BIOINFO E GAUSS. O uso de memória representa o pico de memória (em *gigabytes*) alocada para o Wedring.

CLUSTER	TEMPO DE EXECUÇÃO (HH:MM:SS)	USO DE MEMÓRIA
Bioinfo	04:03:44	1,21
Gauss	03:45:13	1,11

FONTE: O autor, 2013.

Dentre as etapas do Wedring, a etapa de pré-processamento é a mais rápida, já que levou menos de um segundo para ser executada. As etapas de indexação e de expressão diferencial gastaram menos de dois segundos para serem executadas (TABELA 9).

TABELA 9 – TEMPO DE EXECUÇÃO DO *PIPELINE* WEDRING POR ETAPA.

CLUSTER	TEMPO DE EXECUÇÃO POR ETAPA (HH:MM:SS)			
	PRÉ-PROCESSAMENTO	INDEXAÇÃO	MAPEAMENTO	EXPRESSÃO DIFERENCIAL
Bioinfo	< 00:00:01	00:00:13	04:02:03	00:01:27
Gauss	< 00:00:01	00:00:09	03:44:26	00:00:37

FONTE: O autor, 2013.

Por outro lado, a etapa mapeamento é a que consome mais tempo para ser executada (TABELA 9), ocupando mais de 99% do tempo da execução total do *pipeline* em ambos os *clusteres*, uma vez que é nessa etapa que os dados gerados pelo sequenciador são processados, que para o caso de *A. brasilense* fp2 corresponde a 10,5 *gigabytes* de espaço em disco para os arquivos de leitura e qualidade. Além disso, pelo fato de se ter utilizado uma *thread* na configuração do *pipeline*, essa etapa é executada em série, ou seja, os dados de uma biblioteca são processados somente após a conclusão da execução da biblioteca anterior.

O uso de mais de uma *thread* causa a paralelização da etapa de mapeamento, reduzindo o tempo de execução total do *pipeline*. A TABELA 10 apresenta os tempos de execução da etapa de mapeamento para cada biblioteca de *A. brasilense* fp2. Desse modo, paralelizando-se essa etapa seu tempo de execução

seria reduzido a pouco mais de uma hora. Porém, essa redução do tempo é acompanhada pelo aumento do uso de memória alocada, já que os processos da etapa de mapeamento requisitam quantidades de memória para si concomitantemente.

TABELA 10 – TEMPO DE EXECUÇÃO DA ETAPA DE MAPEAMENTO DE PIPELINE WEDRING POR BIBLIOTECA DE *A. brasilense* fp2.

BIBLIOTECA	*CLUSTER*	
	BIOINFO	GAUSS
C1	00:49:20	00:31:55
C2	01:10:25	00:54:32
T1	01:00:42	01:08:20
T2	00:16:58	00:17:53
T3	00:44:38	00:51:46

FONTE: O autor, 2013.

Uma vez que o processamento do Wedring se concentra principalmente na etapa de mapeamento, interrompê-la pode ser problemático, ainda mais se a quantidade de dados analisada for muito grande, pois implica em um tempo de execução considerável. Como solução, há os arquivos de *log* que indicam o estado de execução do *pipeline* até o momento da interrupção. Assim, o usuário pode verificar quais bibliotecas já haviam sido mapeadas e continuar a etapa de mapeamento das que não foram e as outras etapas utilizando os parâmetros de execução parcial (*--just-map*, *--just-counttable* e *--just-de*).

Como resultado do processamento do Wedring sobre os 9,4 *megabytes* dos arquivos de genoma e anotação e 10,5 *gigabytes* das bibliotecas de *A. brasilense* fp2, gerou-se 1,6 *gigabytes* de dados (TABELA 11), o que inclui os arquivos de mapeamento e seus índices, os arquivos de cobertura, o índice de Burrows-Wheeler, os arquivos de *log*, o arquivo com as características gênicas e os dados de expressão gênica.

A maioria desses arquivos ocupa pouco espaço em disco (menos de duas dezenas de *megabytes*), portanto podem ser facilmente transferidos entre computadores (por exemplo, via tráfego de rede). Contudo, os arquivos de mapeamento possuem tamanhos na ordem de centenas de *megabytes*, não sendo uma tarefa trivial movimentá-los entre computadores. Para tanto, é necessário a

utilização de algoritmos de compressão que possam reduzir o tamanho dos arquivos sem que haja perda de dados. Em sua biblioteca padrão, Python possui recursos para lidar com compressão de dados nos formatos gzip, bzip2 e zip (ROSSUM, 2012). Além disso, caso haja a necessidade de conexão entre computadores remotos, há a biblioteca paramiko[13], a qual realiza comunicação segura via rede, tal como a transferência de arquivos.

TABELA 11 – ESPAÇO OCUPADO EM DISCO PELOS ARQUIVOS GERADOs PELO PROCESSAMENTO DE WEDRING SOBRE OS DADOS DE *A. brasilense* fp2. Devido ao tamanho variado dos arquivos, optou-se por utilizar-se de várias escalas de quantidades de bytes para se representar o espaço ocupado em disco, ao invés de uma unidade unificada.

ARQUIVOS	TAMANHO
Mapeamentos no formato BAM	1,5 *gigabytes*
Índices dos mapeamentos	88 *kilobytes*
Índice de Burrows-Wheeler	16 *megabytes*
Coberturas dos mapeamentos	13 *megabytes*
Características gênicas	2,2 *megabytes*
Dados de expressão	628 *kilobytes*
Log	64 *kilobytes*
Total	1,6 *gigabytes*

FONTE: O autor, 2013.

[13] http://www.lag.net/paramiko/

6 CONCLUSÃO

O presente trabalho discorreu sobre o desenvolvimento de Wedring, um *pipeline* para a análise de expressão gênica diferencial em experimentos de RNA-Seq.

Wedring agrupa em uma única ferramenta funcionalidades de cinco *softwares* utilizados em bioinformática: Bowtie, TopHat, SAMtools, BEDTools e DESeq. O pipeline é responsável por processar os dados fornecidos pelo usuário, gerenciar a execução dos programas e a comunicação entre eles, reportar erros e disponibilizar os resultados do processamento.

O *pipeline* foi desenvolvido como um pacote da linguagem de programação Python, atribuindo ao Wedring um caráter modular, o que torna possível sua agregação em outros programas implementados em Python e em outros tipos de recursos, como serviços *web*. Além disso, o programa wedr corresponde a uma interface de linha de comando, que torna possível que Wedring seja introduzido em outros fluxos de trabalho.

Ao final da execução do *pipeline* ficam disponíveis os arquivos de mapeamento no formato BAM e os índices dos mapeamentos, os quais podem ser utilizados em softwares que possuem a funcionalidade de visualização de mapeamentos.

7 PERSPECTIVAS FUTURAS

Wedring é um *pipeline* que possui a função elementar de reportar quais genes são diferencialmente expressos a partir de experimentos de RNA-Seq. Entretanto, certas características ainda podem ser adicionadas ao Wedring para que suas funcionalidades sejam incrementadas.

A etapa de pré-processamento pode ser ampliada para que a validação dos dados de entrada seja mais exigente. Assim, pode-se incluir: (1) mais passos na validação do arquivo de anotação, como a verificação dos valores válidos para cada campo; e (2) a comparação do nome da sequência de referência presente no arquivo de anotação e do nome presente no arquivo do genoma, já que esses necessitam ser idênticos para que os arquivos de cobertura possam ser gerados adequadamente.

Para a etapa de mapeamento há a possibilidade de se substituir o programa BEDTools por um módulo interno ao Wedring que calcule a cobertura do mapeamento, para que as dependências para o funcionamento do Wedring sejam reduzidas. Contudo, essa mudança traz a implicação de ser necessário processar os arquivos de mapeamento no formato BAM, o que pode não ser uma alternativa viável, a não ser que também sejam incluídas bibliotecas para tal.

Outros testes ainda necessitam ser realizados, como a comparação de tempo de execução e uso de memória em relação a outros *pipeline* semelhantes ao Wedring. Além disso, é necessário verificar o comportamento do *pipeline* com o uso de arquivos de leitura em outros formatos, como Fastq, e com genomas eucarióticos.

Para que o uso do *pipeline* seja facilitado, principalmente para usuários não habituados a programas com interface em linha de comando, uma interface gráfica poderia ser elaborada. Outrossim, Wedring poderia ser descrito como um serviço *web*, de tal modo a disponibilizar suas funcionalidades remotamente. Assim, Wedring poderia ser utilizado por meio de um website ou mesmo por dispositivos móveis, como *tablets* e *smartphones*.

REFERÊNCIAS

ADAMS, M. D. et al. Complementary DNA Sequencing: Expressed Sequence Tags and Human Genome Project. **Science**, v. 252, p. 1651-6, 1991.

ALBA, R. et al. ESTs, cDNA microarrays, and gene expression profiling: tools for dissecting plant physiology and development. **The Plant journal : for cell and molecular biology**, v. 39, n. 5, p. 697-714, 2004.

AMERICAN NATIONAL STANDARD FOR INFORMATION SYSTEMS. Coded Character Sets - 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII). ,1986. Nova Iorque: American National Standards Institute.

ANDERS, S.; HUBER, W. Differential expression analysis for sequence count data. **Genome biology**, v. 11, n. 10, p. R106, 2010.

AUER, P. L.; DOERGE, R. W. Statistical design and analysis of RNA sequencing data. **Genetics**, v. 185, n. 2, p. 405-16, 2010.

BENJAMINI, Y.; HOCHBERG, Y. Controlling the False Discovery Rate : A Practical and Powerful Approach to Multiple Testing. **Journal of the Royal Statistical Society**, v. 57, n. 1, p. 289-300, 1995.

BERGER, M. F. et al. Integrative analysis of the melanoma transcriptome. **Genome Research**, v. 20, p. 413-427, 2010.

BULLARD, J. H.; PURDOM, E.; HANSEN, K. D.; DUDOIT, SANDRINE. Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments. **BMC bioinformatics**, v. 11, p. 94, 2010.

BURROWS, M.; WHEELER, D. J. A Block-sorting Lossless Data Compression Algorithm. **SRC Research Report**, 1994.

CAMARENA, L.; BRUNO, V.; EUSKIRCHEN, G.; POGGIO, S.; SNYDER, M. Molecular mechanisms of ethanol-induced pathogenesis revealed by RNA-sequencing. (C. R. Roy, Ed.)**PLoS pathogens**, v. 6, n. 4, p. e1000834, 2010. Public Library of Science.

COCK, P. J. A; FIELDS, C. J.; GOTO, N.; HEUER, M. L.; RICE, P. M. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. **Nucleic Acids Research**, v. 38, n. 6, p. 1767-71, 2010.

DENOEUD, F. et al. Annotating genomes with massive-scale RNA sequencing. **Genome biology**, v. 9, n. 12, p. R175, 2008.

DURBIN, R.; HAUSSLER, DAVID. GFF (General Feature Format) specifications document. Disponível em: <http://www.sanger.ac.uk/resources/software/gff/spec.html>Acesso em: 31/1/2013.

EWING, B.; GREEN, P. Base-Calling of Automated Sequencer Traces Using Phred . II . Error Probabilities. **Genome Research**, v. 8, p. 186-194, 1998.

EWING, B.; HILLIER, L.; WENDL, M. C.; GREEN, P. Base-Calling of Automated Sequencer Traces Using Phred . I . Accuracy Assessment. **Genome Research**, v. 8, p. 175-185, 1998.

FALKNER, J. A; HILL, J. A; ANDREWS, P. C. Proteomics FASTA archive and reference resource. **Proteomics**, v. 8, n. 9, p. 1756-7, 2008.

FALKNER, J. A.; ULINTZ, P. About Fasta. Disponível em: <https://proteomecommons.org/tranche/examples/proteomecommons-fasta/fasta.jsp>Acesso em: 2/2/2013.

FERRAGINA, P.; MANZINI, G. Opportunistic Data Structures with Applications. ,2000. Disponível em: <http://people.unipmn.it/manzini/papers/focs00draft.pdf>.

FONSECA, N. A.; RUNG, J.; BRAZMA, A.; MARIONI, J. C. Tools for mapping high-throughput sequencing data. **Bioinformatics (Oxford, England)**, v. 28, n. 24, p. 3169-3177, 2012.

GAO, D. et al. A survey of statistical software for analysing RNA-seq data. **Human genomics**, v. 5, n. 1, p. 56-60, 2010.

GENTLEMAN, R. et al. Bioconductor: Open software development for computational biology and bioinformatics. **Genome Biology**, v. 5, p. R80, 2004. Disponível em: <http://genomebiology.com/2004/5/10/R80>.

GREGG, C. et al. High-resolution analysis of parent-of-origin allelic expression in the mouse brain. **Science (New York, N.Y.)**, v. 329, n. 5992, p. 643-8, 2010.

HILLIER, L. W. et al. Whole-genome sequencing and variant discovery in C . elegans. **Nature Methods**, v. 5, n. 2, p. 183-188, 2008.

HOLT, R. A; JONES, S. J. M. The new paradigm of flow cell sequencing. **Genome research**, v. 18, n. 6, p. 839-46, 2008.

HUANG, D. W. et al. DAVID Bioinformatics Resources: expanded annotation database and novel algorithms to better extract biology from large gene lists. **Nucleic acids research**, v. 35, n. Web Server issue, p. W169-75, 2007.

IRIMIA, M.; RUKOV, J. L.; PENNY, D.; ROY, S. W. Functional and evolutionary analysis of alternatively spliced genes is consistent with an early eukaryotic origin of alternative splicing. **BMC Evolutionary Biology**, v. 7, p. 188, 2007.

KENT, W. J. et al. The Human Genome Browser at UCSC. **Genome Research**, v. 12, n. 6, p. 996-1006, 2002.

LANGMEAD, B.; TRAPNELL, C.; POP, M.; SALZBERG, S. L. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. **Genome biology**, v. 10, n. 3, p. R25, 2009.

LAWLEY, B.; SIMS, I. M.; TANNOCK, G. W. Whole-transcriptome shotgun sequencing (RNA-seq) screen reveals upregulation of cellobiose and motility operons of Lactobacillus ruminis L5 during growth on tetrasaccharides derived from barley β-glucan. **Applied and environmental microbiology**, v. 79, n. 18, p. 5661-9, 2013. Disponível em: <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3754151&tool=pmcentrez&rendertype=abstract>Acesso em: 24/5/2014.

LEE, H. C. et al. Bioinformatics tools and databases for analysis of next-generation sequence data. **Briefings in functional genomics**, v. 11, n. 1, p. 12-24, 2012.

LEE, N. H.; SAEED, A. I. Microarrays: an overview. In: E. Hilario; J. Mackay (Eds.); **Methods in Molecular Biology**. 2nd ed., v. 353, p.265-300, 2007. Totowa, NJ: Humana Press.

LI, H. et al. The Sequence Alignment/Map format and SAMtools. **Bioinformatics (Oxford, England)**, v. 25, n. 16, p. 2078-9, 2009.

LI, H.; RUAN, J.; DURBIN, R. Mapping short DNA sequencing reads and calling variants using mapping quality scores. **Genome Research**, v. 18, p. 1851-1858, 2008.

MARGUERAT, S.; BÄHLER, J. RNA-seq: from technology to biology. **Cellular and molecular life sciences : CMLS**, v. 67, n. 4, p. 569-79, 2010.

MARIONI, J. C.; MASON, C. E.; MANE, S. M.; STEPHENS, M.; GILAD, Y. RNA-seq: an assessment of technical reproducibility and comparison with gene expression arrays. **Genome Research**, v. 18, n. 9, p. 1509-17, 2008.

MARTIN, J. A.; WANG, Z. Next-generation transcriptome assembly. **NATURE REVIEWS**, v. 12, p. 671-682, 2011.

METZKER, M. L. Sequencing technologies - the next generation. **Nature reviews. Genetics**, v. 11, n. 1, p. 31-46, 2010. Nature Publishing Group.

MORTAZAVI, A.; WILLIAMS, B. A.; MCCUE, K.; SCHAEFFER, L.; WOLD, B. Mapping and quantifying mammalian transcriptomes by RNA-Seq. **Nature Methods**, v. 5, n. 7, p. 621-8, 2008.

NAGALAKSHMI, U. et al. The Transcriptional Landscape of the Yeast Genome Defined by RNA Sequencing. **Science**, v. 320, n. 5881, p. 1344-1349, 2008.

NCBI. National Center for Biotechnology Information. Disponível em: <http://www.ncbi.nlm.nih.gov/BLAST/blastcgihelp.shtml>Acesso em: 28/1/2013.

OKONIEWSKI, M. J.; MILLER, C. J. Hybridization interactions between probesets in short oligo microarrays lead to spurious correlations. **BMC bioinformatics**, v. 7, p. 276-290, 2006.

OSHLACK, A.; ROBINSON, M. D.; YOUNG, M. D. From RNA-seq reads to differential expression results. **Genome biology**, v. 11, n. 12, p. 220, 2010.

OZSOLAK, F.; MILOS, P. M. RNA sequencing: advances, challenges and opportunities. **Nature reviews. Genetics**, v. 12, n. 2, p. 87-98, 2011. Nature Publishing Group.

PARKHOMCHUK, D. et al. Transcriptome analysis by strand-specific sequencing of complementary DNA. **Nucleic acids research**, v. 37, n. 18, p. e123, 2009.

PARKINSON, J.; BLAXTER, M. Expressed Sequence Tags: an overview. In: J. Parkinson (Ed.); **Expressed Sequence Tags (ESTs): Generation and Analysis**. v. 533, p.1-12, 2009. Humana Press.

PEARSON, W. R.; LIPMAN, D. J. Improved tools for biological sequence comparison. **Proceedings of the National Academy of Sciences of the United States of America**, v. 85, n. 8, p. 2444-8, 1988.

PICARDI, E. et al. Large-scale detection and analysis of RNA editing in grape mtDNA by RNA deep-sequencing. **Nucleic acids research**, v. 38, n. 14, p. 4755-67, 2010.

QUINLAN, A. R.; HALL, I. M. BEDTools: a flexible suite of utilities for comparing genomic features. **Bioinformatics (Oxford, England)**, v. 26, n. 6, p. 841-2, 2010.

R CORE TEAM. R: A language and environment for statistical computing. ,2012. Vienna, Austria: R Foundation for Statistical Computing. Disponível em: <http://www.r-project.org/>.

ROBINSON, J. T. et al. Integrative genomics viewer. **Nature biotechnology**, v. 29, n. 1, p. 24-26, 2011.

ROBINSON, M. D.; SMYTH, G. K. Moderated statistical tests for assessing differences in tag abundance. **Bioinformatics (Oxford, England)**, v. 23, n. 21, p. 2881-7, 2007.

ROSSUM, G. VAN. Python v2.6.8 documentation. Disponível em: <http://docs.python.org/2.6/index.html#>.

ROYCE, T. E.; ROZOWSKY, J. S.; GERSTEIN, M. B. Toward a universal microarray: prediction of gene expression through nearest-neighbor probe sequence identification. **Nucleic acids research**, v. 35, n. 15, p. e99, 2007.

SANGER, F.; NICKLEN, S.; COULSON, A. R. DNA sequencing with chain-terminating inhibitors. **Proceedings of the National Academy of Sciences of the United States of America**, v. 74, n. 12, p. 5463-5467, 1977.

SILVEIRA, L. D. **MONTAGEM E ANOTAÇÃO PARCIAL DA SEQUÊNCIA GENÔMICA DA BACTÉRIA DIAZOTRÓFICA Azospirillum brasilense FP2**, 2012. Universidade Federal do Paraná.

TANIGUCHI, M.; MIURA, K.; IWAO, H.; YAMANAKA, S. Quantitative assessment of DNA microarrays--comparison with Northern blot analyses. **Genomics**, v. 71, n. 1, p. 34-9, 2001.

THE GENE ONTOLOGY CONSORTIUM et al. Gene Ontology : tool for the unification of biology. **Nature Genetics**, v. 25, n. 1, p. 25-29, 2011.

THE SAM FORMAT SPECIFICATION WORKING GROUP. The SAM Format Specification (v1.4-r985). , p. 1-11, 2011.

THORVALDSDÓTTIR, H.; ROBINSON, J. T.; MESIROV, J. P. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. **Briefings in bioinformatics**, p. 1-15, 2012.

TRAPNELL, C. et al. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. **Nature biotechnology**, v. 28, n. 5, p. 511-5, 2010. Nature Publishing Group.

TRAPNELL, C.; PACHTER, L.; SALZBERG, S. L. TopHat: discovering splice junctions with RNA-Seq. **Bioinformatics (Oxford, England)**, v. 25, n. 9, p. 1105-11, 2009.

WANG, E. T. et al. Alternative Isoform Regulation in Human Tissue Transcriptomes. **Nature**, v. 456, n. 7221, p. 470-476, 2008.

WANG, Z.; GERSTEIN, M.; SNYDER, M. RNA-Seq: a revolutionary tool for transcriptomics. **Nature reviews. Genetics**, v. 10, n. 1, p. 57-63, 2009.

WILHELM, B. T.; LANDRY, J.-R. RNA-Seq—quantitative measurement of expression through massively parallel RNA-sequencing. **Methods**, v. 48, p. 249-257, 2009.

YANG, Y. H. et al. Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. **Nucleic acids research**, v. 30, n. 4, p. e15, 2002.

YENDREK, C. R.; AINSWORTH, E. A; THIMMAPURAM, J. The bench scientist's guide to statistical analysis of RNA-Seq data. **BMC Research Notes**, v. 5, n. 1, p. 506, 2012. BMC Research Notes.

YOUNG, M. D.; WAKEFIELD, M. J.; SMYTH, G. K.; OSHLACK, A. Gene ontology analysis for RNA-seq: accounting for selection bias. **Genome biology**, v. 11, n. 2, p. R14, 2010.

APÊNDICE

APÊNDICE 1 – *SCRIPT* EM R PARA CRIAÇÃO DE GRÁFICOS PARA ANÁLISE DOS VALORES DE QUALIDADE DAS LEITURAS.

```
args <- commandArgs(trailingOnly=TRUE)

adjustOutFileName <- function(out.dir, prefix, qualifier, label) {
  file.name <- paste0(paste(prefix, qualifier, label, sep='_'), '.jpg')
  return(file.path(out.dir, file.name))
}

qual.file <- args[1]
label <- ifelse(is.na(args[2]), '', args[2])
qualifier <- ifelse(is.na(args[3]), '', args[3])
out.dir <- ifelse(is.na(args[4]), '.', args[4])

t <- read.table(qual.file, comment.char='>')
categories <- seq(length(t))

jpeg(adjustOutFileName(out.dir, 'qual_per_pos', qualifier, label),
     width=1500, height=525)

# Box-plot do valor de qualidade de cada posição das leituras.
# Um gráfico de linha na cor vermelha também é plotado ligando os valores
# de mediana de cada posição.
boxplot(t, names=categories,
        main=paste('Qualities per read position -', label, qualifier),
        xlab='Read position', ylab='Quality values',
        xlim=c(1, 50), ylim=c(-1, 35))
lines(categories, apply(t, 2, median), col='red', lwd=2)
dev.off()

jpeg(adjustOutFileName(out.dir, 'qual_dist', qualifier, label), width=840,
     height=525)

# Histograma da distribuição dos valores de qualidade.
hist(as.vector(as.matrix(t)), seq(-1, 35),
     main=paste('Distribution of quality values -', label, qualifier),
     xlab='Quality values', ylab='Frequency', col='blue3', border='white')
dev.off()
```

ANEXO

ANEXO 1 – SEQUENCIAMENTO PELA PLATAFORMA LIFE TECHNOLOGIES SOLID. Fonte: Pandey, Nutter e Prediger (2008)[14]

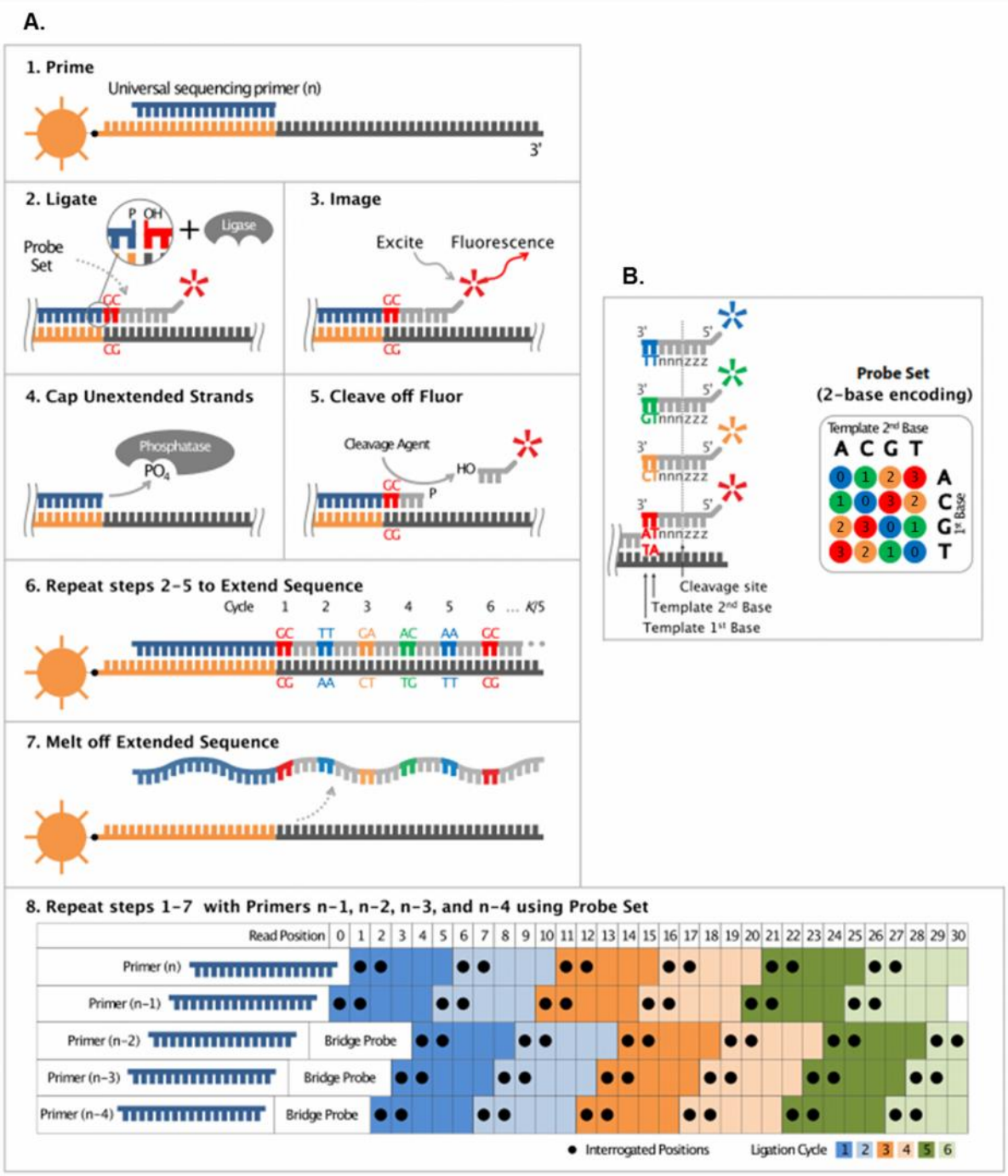
O sistema *Life Technologies* SOLiD é uma tecnologia de sequenciamento de alta produção e acurácia. Essa tecnologia se baseia no sequenciamento paralelo massivo de fragmentos amplificados de DNA, utilizando-se da ligação sequencial de nucleotídeos marcados por fluorescência.

A primeira etapa dessa técnica é a preparação da biblioteca de fragmentos que será sequenciada, podendo-se gerar bibliotecas *single-end* ou *paired-end*. Essa etapa envolve a extração e fragmentação do material genético a ser analisado, como DNA ou mRNA, seguido ligação de adaptadores (P1 e P2) nas extremidades do fragmento e da conversão para cDNA.

Então, a biblioteca é amplificada por PCR de emulsão, pelo qual os fragmentos são ligados a microesferas magnéticas. Após uma etapa de purificação, as microesfera ligadas aos fragmentos amplificados são depositadas em lâminas de vidro quimicamente modificadas para aderir-se covalentemente às microesferas por meio dos adaptadores P2. Essas lâminas são componentes da plataforma SOLiD e é sobre elas que ocorre a reação do sequenciamento.

O sequenciamento (PAINEL I) procede iniciando-se pela ligação de um *primer* universal ao adaptador P1. Desse modo a sequência desconhecida do fragmento é interrogada por sondas de oito nucleotídeos marcados com fluorescência. O conjunto de sondas é composto por todas as combinações de A, T, C e G nas posições um a cinco. Somente a sonda que é homóloga as primeiras cinco bases do fragmento é capaz de se ligar adjacentemente ao *primer* universal. Essa ligação é feita por uma enzima ligase e como resultado um sinal de fluorescência é emitido. Esse sinal é emitido em quatro cores diferentes, dependendo das bases presentes na posição um e dois da sonda.

[14] PANDEY, V.; NUTTER, R. C.; PREDIGER, E. (2008) Applied Biosystems SOLiD™ System: Ligation-Based Sequencing, **in Next Generation Genome Sequencing**: Towards Personalized Medicine (ed M. Janitz), Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim, Germany.



PAINEL 1 – REAÇÃO DE SEQUENCIAMENTO BASEADA EM LIGAÇÃO. (A) Um primer universal complementar ao adaptador P1 é ligado ao fragmento. Sondas marcadas com fluorocromos são ligados ao fragmento, emitindo um sinal luminoso de uma cor específica. A sonda é clivada na posição seis e uma nova sonda é inserida. Esse processo é repetido de quatro a seis vezes. O ciclo se reinicia utilizando-se primers de tamanhos de uma a quatro bases menores que o primer inicial. (B) Codificação em duas bases utilizada pelas sondas e códigos numéricos equivalentes.
FONTE: Modificado de Applied Biosystems, 2011[15].

[15] APPLIED BIOSYSTEMS. **SOLiD System accuracy with the Exact Call Chemistry module**. Disponível em: <http://www3.appliedbiosystems.com/cms/groups/global_marketing_group/documents/generaldocuments/cms_091372.pdf> Acesso em: 13/01/2013.

Em seguida, as bases na posição seis da sonda são removidos junto com a marcação fluorescente por um agente de clivagem e, então, uma nova sonda é ligada e um sinal emitido. Esse ciclo é repetido por mais quatro a seis vezes, dependendo do tamanho de leitura desejado.

A sequência gerada tendo-se o fragmento como molde é removida e um novo *primer* com uma base a menos é ligado ao fragmento. E do mesmo modo o processo de ligação das sondas e emissão do sinal é repetido até que se utilize um *primer* com até quatro bases as menos. Assim, todas as bases do fragmento são interrogadas ao menos duas vezes.

A esse processo de sequenciamento dá-se o nome de “Codificação em Duas Bases” (“*2-Base Encoding*”), devido à cor de o sinal emitido depender da combinação de bases presentes nas posições um e dois das sondas. Como resultado do sequenciamento, as sequências dos fragmentos (junto com seus valores de qualidade) são reportadas em espaço de cor, no qual os valores 0, 1, 2 e 3 são utilizados para representar a cor do sinal emitido pela sonda.