# Capítulo 4. Montagem de genomas: do *de novo* à finalização *in silico*

Frederico Schmitt Kremer 1, Luciano da Silva Pinto 1

‘ *Laboratório de Bioinformática e Proteômica, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas*

**Objetivos:** Apresentar os princípios da montagem *de novo* de genomas, metologias disponíveis para o processo de *scaffolding* e técnicas para finalização de montagens através de ferramentas *in silico*.

As plataformas de NGS são capazes de gerar um grande volume de dados, o que torna possível o sequenciamento rápido de genomas e transcriptomas inteiros, além de possuirem diversas outras aplicações, como metagenômica e detecção de variantes genéticas. Para que uma amonstra de DNA seja sequenciada por estas plataformas, é necessário se realizar a sua fragmentação, seguida da deleção dos fragmentos que possuem tamanho compatível com as especificações do equipamento.

O processo de fragmentação deve ser aleatório, de forma a permitir fragmentos “sobreponíveis” sejam gerados. Diferentes abordagens podem ser utilizadas para esta finalidade, o que inclui **métodos físicos**, como uso de temperatura (usada para RNA), nebulização e sonicação, e **métodos químicos**, com o uso de combinações de enzimas de restrição. Após ser fragmentado, os fragmentos que apresentam o tamanho no intervalo de interesse são selecionados, normalmente através de eletroforese.

Após a fragmentação, e das demais etapas de preparo da biblioteca de sequencimaneto, as sequencias de DNA serão lidas pelo sequenciador e resultaram em milhões (ou até bilhões) de leituras de sequenciamento (*reads*). As leituras resultantes poderão ser pré-processadas, como visto no capítulo anterior, e posteriormente precisarão ser montadas para que as sequencias de DNA presentes na amostra original sejam re-construidas.

## Montagem de sequências

As abordagens utilizadas para a montagem de uma sequência podem ser classificadas em duas caterogorias: **montagem *de novo*** e *montagem guiada por referência*. No presente capítulo nos focaremos na montagem de novo, sendo a montagem por referência abordada nos capítulos posteriores, quando falarmos em **análise de variantes** e **análise de dados de RNA-Seq**.

**Montagem *de novo*:** A montagem *de novo* é realizada através da identificação de sobreposição entre as leituras, seguida da geração de *contigs* pela identificação de uma, ou várias, sequencia(s) consenso. O nome “*de novo*” se refere a ausência de uma informação prévia, em contrate com a *montagem guiada por referência*.

**Montagem guiada por referência:** A montagem guiada por referência utilizada as informações de uma sequência conhecida (referência) para a reconstrução da sequencia de interesse. Neste caso, as leituras do sequenciamento não são comparadas entre si, mas sim alinhadas contra a referência, sendo o resultado da sobreposição dos alinhamentos usado para a construção de uma sequência concenso.

A montagem *de novo* (do latim: “do início”) consiste na reconstrução da sequência genômica original através da identificação de sobreposição das leituras (ou *reads*), seguida da construção se sequências consenso. Estas sequências consenso são também denominadas *contigs*, visto que são “contiguas”, ou seja, ininterruptas. Posteriormente, várias *contigs* podem ser unicadas em *scaffolds* através de uma determinada informação, para gerar sequências ainda maiores. Ao contrário das *contigs*, as *scaffolds* não são ininterruptas, o que significa que estas podem possuir uma ou várias regiões faltantes que representam falhas (*gaps*) na montagem.

Por padrão, *gaps* são representados com regiões contendo várias repetições da letra “N”, a mesma letra usada para indicar as bases que foram mal-identificadas no sequenciamento. Entretanto, diferentes dos erros de chamada de base, que normalmente afetam apenas um ou alguns nucleotídeos, os gaps de montagem costumam possuir dezenas ou centenas de “N”s.

### Algoritmos para montagem *de novo*

*Algoritmos exaustivos*

Um algoritmo é considerado exaustivo quando a sua execução considera todas as possibilidades dentre de um dado “espaço de busca”. No caso da montagem de sequências, por exemplo, um algoritmo exaustivo realizaria o alinhamento de todas as sequência contra todas antes de identificar as melhores sobreposições para efetuar a montagem. Como todas as possibilidades são exploradas, é esperado que este método resulte um uma montagem mais próxima da “solução ótima” para os dados de interesse.

Os algoritmos exaustivos são muitas vezes chamados “algoritmos gulosos”, ou gredy, e isso se refleta no tempo de execução (e muitas vezes também no consumo de memória), que tende a crescer expressivamente a medida que o número de dados a serem analisados aumenta. Considerando um conjunto de 10 sequencias, por exemplo, um algoritmo de montagem *gredy* precisaria analisar sobreposições entre 45 pares de sequências.

**Obs:** 45 combinações pois neste caso estamos lidando com uma análise combinatória, e a intenção é uma conbinação não um arranjo, já que a orderm da comparação não importa. Neste caso, são um total (n) de 10 itens de grupos (s) de 2, e o calculo do número de combinações é feito através da formula:



Desta forma, é fácil entender por que este tipo de algoritmo não é viável de ser utilizado quando se tem milhões de sequencias para serem montadas. Caso tenhamos 100 sequencias, o número de alinhamos de sobreposiçõe a serem computados já sobe para 4950, e é praticamente impossível calcular o número total de alinhamentos caso sejam consideradas 100.000.000 de sequência, e muitos menos executar todos estes em um intervalo de tempo hábil. Desta forma, os algoritmos *gredy* deixaram de ser aplicados na montagem de genomas obtidos com plataformas de nova geração, sendo substituidos por **abordagens heurísticas** que reduzem o espaço de busca.

*Algoritmos do tipo OLC (Overlap-Layout-Consensus)*

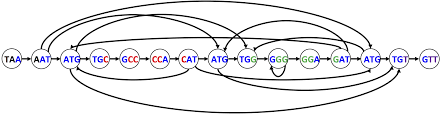
O modelo *overlap-layout-consensus* consiste em: realizar um alinhamento de todas as leituras contra todas as leituras, sendo similares aos algoritmos *gredy*, de forma a se identificar as regiões de sobreposição (*overlap*); construção de um grafo de leituras, que representa as conexões identificadas entre as leituras e permite o agrupamento das sequencias (*layout*); construção de uma conjunto de sequencias consenso através da exploração dos caminhos presentes dentro do grafo. Neste algoritmo, as sequencias das leituras são utilizadas explicitamente durante a construção das *contigs*. Durante o alinhamento “todas contra todas”, no entando, é possível se reduzir o tempo necessário para cada comparação através da busca que regiões similares apenas nas extremidades das sequencias, ou de outras técnicas heurísticas.

*Algoritmos baseados em grafos de Bruijn*

Os grafos *de Bruijn* foram desenvolvidos como uma alternativa ao modelo *overlap-layout-consensus* e sobretudo para antender as demandas que surgiram com as plataformas de sequenciamento Illumina Solexa e ABI SOLiD. Estes algoritmos recebem este nome pois tem sua base nos trabalhos de Nicolaas Govert de Bruijn, que fundamentou o uso de *k-mers* para a reconstrução de sequências de texto (*strings*). Neste caso, k-mers são sub-fragmentos sobreponíveis de tamanho *k*, que serão derivados de um determinando conjunto de fragmentos que se pretende montar. No caso de uma montagem de genoma, por exemplo, os fragmentos são as leituras de sequenciamento.

O primeiro passo para se realizar a montagem a partir desta abordagem é se extrair os *k-mers* de todas se leituras de interesse, e transformar-los nos nodos do grafo. Posteriormente, os nodos que que tem sobreposições exatadas de *k-1* nucleotídeos são conectados, o que permite a identificação das regiões de sobreposição. Para se construir as contigs, os algoritmos exploram as conexões presentes no grafo, passando pelos k-mers até achar o melhor conjunto de caminhos indepententes.

Para a montagem de genomas é comum se usar k-mers com tamanho > 20 nucleotídeos, mas consideravelmente menores o tamanho das leituras. Também é geralmente exigido o uso de um numero ímpar como valor que k, uma vez que sequencias com tamanho par apresentam maior propensão a serem palindrômicas, o que pode confundir o algoritmo de montagem duranta a exploração do grafo.



## Cobertura de uma montagem

A cobertura de uma montagem diz respeito a quantas vezes cada base foi sequenciada, podendo ser medida “base-a-base”, ou na forma de uma “cobertura média”. Valores de cobertura são usualmente representados na forma de multiplicações (ex: “30x”, que indica que uma cobertura de 30 vezes).

No caso do sequenciamento de nova geração, de forma a se reduzir a quantidade de memória utilizada pelo programa de montagem, muitos algoritmos, mas sobretudo os baseados em gragos *de Bruijn*, não guardam a informação de quais leituras foram utilizadas para a construção de cada *contig* (*read-tracking*). Entretanto, é possível se inferir esta informação através do re-mapeamento das leituras contra a montagem.

O processo de re-mapeamento das leituras contra a montagem apresenta algunas vantagens em relação ao *read-tracking*, sobretudo no que diz respeito ao uso de memória e à possibilidade de se identificar possíveis erros de montagem acarretados pelos grafos *de Bruijn* com uso de ferramentas auxiliares. Como os programas de usados de re-alinhamento não são baseados no mesmo algoritmo utilizado na montagem, é esperado que existam discrepâncias, sobretudo em regiões repetitivas e / ou quando houverem as leituras contiverem artefatos de sequenciamento. Através dos dados de cobertura *base-a-base* é possível se avaliar sua acurácia na montagem de uma determinada região, uma vez que muitas vezes as sequencias das leituras são também acompanhadas de seus respectivos valores de qualidade. Também é possível se avaliar se uma determinada base foi identificada corretamente. É possível também se calcular a cobertura média de uma montagem, ou mesmo de uma região de interesse, através da média aritmética das coberturas de cada base.

Em alguns casos, sobretudo antes da montagem, é comum se falar em “cobertura esperada” (*expected coverage*). Este valor é normalmente calculado se dividindo o número de bases presentes no arquivo de leituras pelo tamanho esperado do genoma. Neste caso, é considerada uma fragmentação homogênia do genoma, e que todas as leituras presentes no arquivo serão utilizadas para a montagem, o que geralmente não acontece.

### Exemplo de montador *de novo*: Velvet

O Velvet é um programa de montagem *de novo* de genomas baseado em grafos *de Bruijn*, sendo sido um dos primeiros a permitir a montagem de dados sequenciado com uso das plataformas Illumina Solexa*.* Apesar de já ultrapassado, o Velvet ainda é bastante usado, sendo também parte outros programas de montagem como o IMAGE (fechamento de *gaps*) e o OASES (montagem de transcriptomas). O pacote Velvet é composto pelos subprogramas velveth (responsável pelo processamento dos arquivos de entrada e inico da construção do grafo) e velvetg (responsável pela re-construção das *contigs*), e pode ser instalado através do gerenciador de pacotes apt com uso da linha de comando:

$ sudo apt-get install velvet

Considerando um arquivo “dataset.fastq” que contêm dados de sequenciamento de um genoma de interesse, é possível se realizar a sua montagem através dos seguintes comandos.

$ velveth montagem 31 –short –fastq dataset.fastq

$ velvetg montagem

Neste caso, o programa velveth do pacote velvet realiza o preparo do grafo a partir do arquivo gerado, usando um tamanho de *k-mer* de 31 nucleotídeos e salvando tudo na pasta “montagem”. As opções “-short” indicam que as leituras são do tipo short-read, e o “-fastq” indica o formato do arquivo. Posteriormente, o programa velvetg faz exploração da grafo, e salvo o resultados na mesma pasta, sendo as *contigs* salvas no arquivo “contigs.fa”.

## *Scaffolds*

Como dito anteriormente, *contigs* são sequencias montadas a partir da sobreposição de leituras, sendo ininterruptas, ou seja, não contendo *gaps*. Estas sequencias podem, no entanto, ser conectadas através de informações adicionais, denominadas “informações de ligação” (*linkage information*), dando origem as *scaffolds*. Diferentes informações de ligação podem ser utilizadas para para a geração de *scaffolds,* sendo as leituras pareadas e os genomas de referência algumas das mais utilizadas atualmente.

As *Scaffolds* consitem em duas ou mais *contigs* que são conectadas por uma região desconhecida, denominada *gap* de montagem, e que posuem uma orientação e posição relativa conhecida ou ao menos estimada. O tamanho do *gap* entre as *contigs* de uma *scaffolds* pode ter um tamanho estimado ou arbitrário, dependendo da informação que foi utilizada para a sua construção.

### Geração de *scaffolds* através de leituras pareadas

Algumas plataformas de sequenciamento permite a geração das chamadas leituras-paradas, que consistem em pares *reads* que são derivadas das extremidades de um mesmo fragmento de DNA. As leituras pareadas podem ser de dois tipos: *paired-end* e *mate-pairs*. Leituras do tipo *paired-end*, como as geradas pelas plataformas Illumina e Roche 454, costumam ser derividas de fragmentos de DNA relativamente curtos, onde muitas vezes o espaço entre cada leitura é menor que 1 kb. Por outro lado, as bibliotecas do tipo *mate-pair* possuem espaçamentos muito maiores, podendo as extremidades das leituras apresentarem uma distância de mais de 10 kb.

As leituras geradas a partir de uma mesmo fragmento recebem uma mesma identificação, o que permite que esta informação seja explorada durante o processo de montagem do genoma. Em alguns casos, por exemplo, é possível que durante a construção do grafo *de Bruijn* as leituras derivada de uma das extremidades tenha sido usada na construção de separadas. Nesa situação, caso haja um número suficiente de leituras apontando para a existência desta conexão, é possível se unir estas *contigs* em uma *scaffold*.

### Geração de *scaffolds* através de leituras longas

As leituras longas, como as geradas a partir das plataformas PacBio e Oxford Nanopore, tendem a apresentar uma alta taxa de erro quando usadas isoladamente, mas podem ser úteis para diversas aplicações, sendo uma delas a geração de *scaffolds*. Este tipo de abordagem segue a mesma lógica da aplicação de leituras pareadas, sendo feito o alinhamento das leituras longas contra as *contigs* e posteriormente se conectando as *contigs* que alinharam contra as mesmas leituras.

### Geração de *scaffolds* com base em um genoma de referência

Apesar da montagem *de novo* não utilizar informações de outros genomas durante a construção das *contigs*, é possível se aproveitar este tipo de informação para a geração de *scaffolds*. Esta abordagem é comum quando já existe genomas de organismos filogenéticamente próximos, sobretudo no caso de bactérias, sendo estes genomas denominados “genomas de referência”. Um genoma de referência é geralmente uma sequência finalizada, ou seja, não está quebrado em *contigs* ou *scaffolds*, mas sim estruturado com cromossomo(s) inteiro(s).

A partir das *contigs* de um genoma de interesse e uma sequencia de referência, basta se realizar o alinhamento entre estes dois conjuntos de sequências e usar os dados do alinhamento para achar a posição e orientação correta das *contigs* dentro de genoma de interesse. A geração de *scaffolds* com base em genomas de referência será abordada mais a fundo posteriormente neste capítulo.

### Exemplo de geração de *scaffolds* com base em um genoma de referência: ABACAS

Uma das ferramentas mais simples para o ordenamento de contigs com base em genomas de referências é o ABACAS. Este ferramenta utilizar como base os programas de alinhamento do pacote Mummer para se alinhar as *contigs* do genoma de interesse contra a sequencia de referência, e gera uma sequência de *scaffold* com base nos dados de alinhamento, sendo também feita a predição do tamanho de cada *gap* com base na cobertura de cada alinhamento. O ABACAS pode ser instalado através do comando:

$ sudo apt-get install abacas

Considerando um arquivo de montagem “meu\_genoma.fasta” e uma referência “ref.fasta”, é possível se realizar o processo de geração da scaffold através do comando:

$ abacas –q meu\_genoma.fasta –r ref.fasta –p nucmer

A opção “-p nucmer” indica que o programa nucmer, do pacote Mummer, será utilizado para a realização do alinhamento. Alternativamente, é possível também se utilizar o programa promer, mas isso resultará em um aumento expressivo no tempo necessário para a computação dos alinhamento, apesar de eventualmente esta opção resultar em uma maior acurácia.

## Fatores que afetam a montagem *de novo*

A qualidade da montagem de um genoma obtido a partir de leituras curtas (short-reads) de NGS é afetada por diversos **fatores intrínsecos** e **extrínsecos**. Dentre os fatores intrísencos podemos incluir as sequencias de DNA repetitivas, como as repetições em tandem (ex: VNTR, SSR, microssatélites) e as sequências parálogas (ex: rRNAs, tRNAs e genes de transposases). Já no caso dos fatores extrínsecos temos as limitações das próprias plataformas de sequenciamento, como por exemplo os erros de homopolímeros.

A dificuldade imposta pelas repetições em tandem e sequencias parálogas se dão sobretudo quando o tamanho da região repetida é muito menor que o tamanho médio das leituras. No caso de repetições em tandem, é possível que um mesma sequencia de alguns nucleotídeos (ex: ‘ATTA’) esteja repetido dezenas ou centenas de vezes. Durante o processo de fragmentação, está região será fragmentada e dará origem a um grande número de leituras iguais, o que possivelmente será mal interpretado pelo programa de montagem. Neste caso, é provável que a região formada na montagem será consideravelmente menor que a real. É possível se detectar este tipo de erro através da comparação da cobertura nestas regiões com o observado ao longo do genoma, mas a montagem exata é computacionalmente impossível como leituras curtas.

De forma similar as repetições em tandem, regiões diferentes do genoma que compartilham sequencias com alta similaridades, muitas vezes derivadas da duplicação gênica (parálogas), também podem resultar em erros de montagem. Neste caso, as leituras compartilhas são geralmente atribuidas como pertencentes a apenas uma das regiões, e as demais regiões acabam fragmentadas em diferentes *contigs*. Em alguns casos, as regiões repetitivas podem ser “mascaradas”, e posteriormente resolvidas através do uso de dados auxiliares, como leituras pareadas, em etapas de *scaffolding* e fechamendo de *gaps*.

No que diz respeito aos fatores extríncecos ao genoma, é possível que erros específicos da plataforma de sequenciamento também afetem a qualidade da montagem gerada. No caso das plataformas 454 e IonTorrent, por exemplo, o erro na identiciação de sequencias homopoliméricas pode afetar a construção das contigs no grafo de Bruijn, uma vez que os k-mers derivados destas regiões podem não apresentar concordância, o que resulta em uma maior fragmentação da montagem. Já no caso da Illumina, uma diminuição na qualidade média das bases localizada na região final das leituras pode resultar em fins abruptos (*dead-ends*) nos caminhos (e consequentemente nas *contigs*) identicados pelo algoritmo. Desta forma, é importante se considerar o uso de diferentes ferramentas de montagem, e dar preferências à aquelas que apresentam maior tolerância aos erros observados na plataforma de interesse.

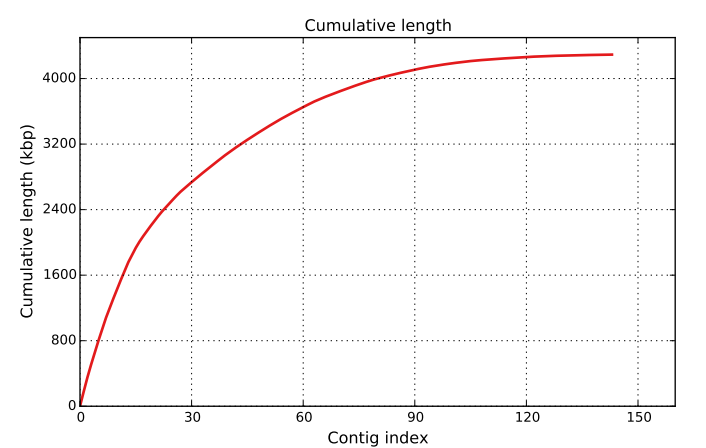
## Avaliação de montagens

### Ao montar um genoma através da estratégia *de novo* é importante se ter ferramentas para avaliar a qualidade de cada montagem. Desta forma, caso existam montagens alternativas para o mesmo organismo, é possível se selecionar a que se mostra menos fragmentada, por exemplo, por meio de determinadas métricas que podem ser calculadas a partir das sequencias das *contigs* / *scaffolds*. Atualmente, a ferramenta mais utilizada para o calculo dessas métricas é o QUAST, que pode ser executado localmente (disponível em <http://quast.sourceforge.net/>) ou através de sua interface *web* (disponível em <http://quast.bioinf.spbau.ru/>).

### Tamanho cumulativo

A montagem *de novo* de um genoma sequenciado através de plataformas de NGS pode resultar em um número variado de *contigs*, cada uma com um tamanho específico. Diversos fatores contribuem podem afetar a distribuição do tamanho das *contigs* dentro de uma montagem, mas é esperado que esta obedeça uma tendência de distribuição logaritma, sendo o desvio desta tendência possivelmente uma consequência de erros de montagem, ou artefatos de sequenciamento. É possível se avaliar a distribuição do tamanho das *contigs* presentes em uma montagem através da análise de tamanho cumulativo.

Para se realizar a análise de tamanho cumilativo é necessário: (a) ordenar as *contigs* por tamanho, das maiores para as menores; (b) atribuir a cada *contig* um índice número, iniciando a partir de 1; (c) passar por cada *contig* da montagem, e se somar o tamanho desta com o das anteriores, sendo este valor denominando “tamanho cumulativo”; (d) gerar um gráfico onde o *x* indica o indice da contig, o *y* o seu respectivo tamanho cumulativo.

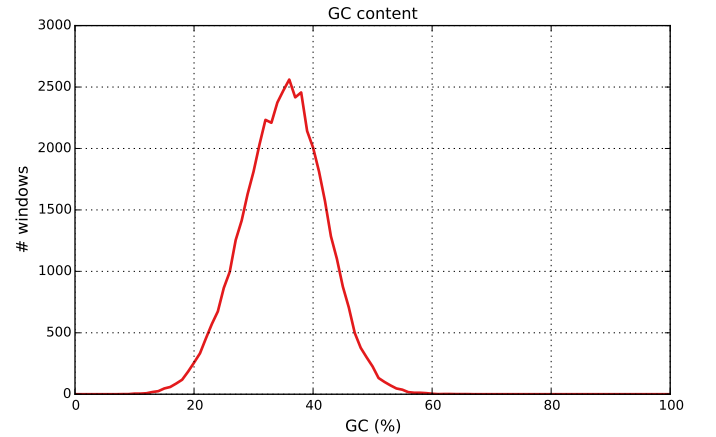


**Figura 1.** Exemplo de uma distribuição de tamanho cumulativo.

## Distribuição de conteúdo CG

O conteúdo CG (CG%) de um genoma consiste na porcentagem de pares de base C-G presentes em relação ao tamanho total das suas sequencias. Ao longo do genoma, diferentes regiões apresentarão valores menores ou maiores que esta média, o que tende a se manifestar a forma de uma **distribuição gausiana** (normal).

Para se calcular a distribuição de CG com base em uma sequencia, muitas vezes os programas utilizam a abordagem de *sliding window*, que consiste em analisar diferentes intervalos sobreponíveis de tamanho fixo, denominada **janela** (*window*) e calcular a porcentagem de CG% para cada janela. Posteriormente, as porcentagens, de 0 a 100, são plotadas no eixo X de um gráfico, e o número de janelas identificas para cada porcentagem são indicados no eixo Y.



**Figura 2.** Exemplo de uma distribuição de tamanho conteúdeo GC.

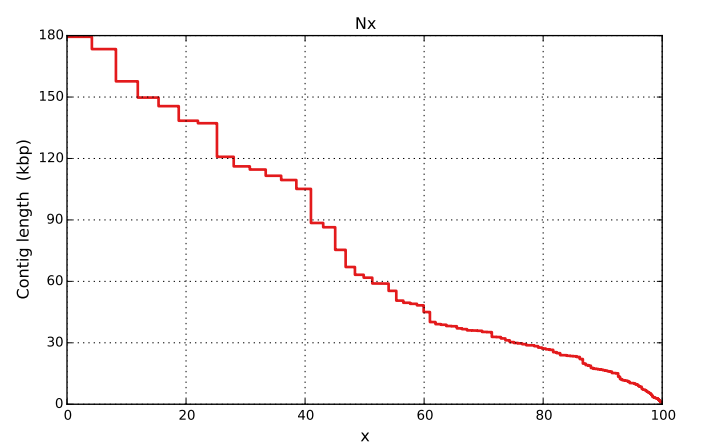
### Nx, Lx e NGx

Diferentes métricas podem ser utilizadas para aferir o quão fragmentada ou “contigua” está a montagem de um genoma, sendo as Nx, NGx e Lx, as mais utilizadas. Neste caso, o “x” é uma referência à porcentagem da montagem que é cobertura por um determinado conjunto de *contigs*.

**Nx (ex: N50, N75, N90):** A expressão“*Nx = y*” presenta que x% das bases presentes na montagem analisada estão contidas em contigs de tamanho *y* ou superior. Ex: N50 = 50.000, significa que 50% das bases presentes na montagem estão contidas em contigs com tamanho igual ou maior que 50.000. Para se calcular esta métrica é necessário: (a) calcular o tamanho total da montagem; (b) ordenar as contigs em order descrente de tamanho; (c) passar de *contig* em *contig*, calculando com base nas contigs anterior o tamanho cumulativo da montagem. Quando o valor obtido representar x% ou mais do tamanho da montagem, o Nx será igual ao tamanho da *contig* atual. É possível também se calcular todos os valores de Nx, com x variando de 0 a 100, e se representar na forma de um gráfico. Ao se comparar diferentes montagens para um mesmo genoma, quanto maior o valor de Nx, menor fragmento a fragmentação.

**NGx (ex: N50, N75, N90):** A métrica NGx é similar à Nx, mas ao invés de utilizar o tamanho da montagem como base, é usado o tamanho esperado do genoma, ou o tamanho previamento conhecio.

**Lx (ex: N50, N75, N90):** Similar ao Nx, mas ao invés de representar o tamanho da *contig*, representa o número de *contigs* necessário para se atingir x% das bases da montagem, ao se ordenar estas em ordem descrecente de tamanho.



**Figura 3.**  Exemplo de representação de distribuição de Nx.

## Integração de diferentes montagens *de novo*

Apesar de boa parte das ferramentas de montagem *de novo* utilizar algoritmos similare para executar o processo “bruto” de montagem, é comum observar resultados ligeiramente diferentes quando um mesmo genoma é montado por diferentes. A principal causa para esta divergência é a forma como estas ferramentas tratam os erros decorrentes do processo de sequenciamento (ex: sequencias homopoliméricas, substituições de base) assim como as regiões problemáticas presentes em alguns genomas (ex: genes duplicados, sequencias de repetições em tandem). Desta forma, é esperada uma maior “complitude” na montagem gerada a partir da integração.

Existem diferentes programas de integração de montagens, sendo alguns baseados em informações de dados de sequenciamento, e outros baseados unicamente em alinhamento entre as montagens alternativas. Um exemplo de ferramenta para este propósito é o CISA,que permite a geração de montagens consenso para um genoma a partir de 3 ou mais montagens alternativas.

### Rodando o CISA

O CISA pode ser obtido a partir do endereço: <http://sb.nhri.org.tw/CISA/en/CISA>. O programa é composto por dois scripts: Merge.py, que realiza a junção dos arquivos das diferentes montagens, e CISA.py, que realizar o processo de integração propriamente dio. Estes dois programas são executadas da mesma forma, através de arquivos de configuração.

Exemplo de arquivo de configuração para o *script* Merge.py: merge.config

count=3

data=RCA\_clc.fasta,title=Contig\_m1

data=RCA\_newbler.fasta,title=Contig\_m2

data=RCA\_spades.fasta,title=Contig\_m3

Master\_file=contigs.merged.fasta

min\_length=100

Gap=11

Exemplo de arquivo de configuração para o *script* CISA.py: cisa.config

genome=4500000

infile=contigs.merged.fasta

outfile=contigs.integrated.fasta

nucmer=nucmer

R2\_Gap=0.95

CISA=/home/cdtec/Frederico/Documentos/CISA1.3/

makeblastdb=makeblastdb

blastn=blastn

Para executar estes *scripts*, a partir dos arquivos de configuração, basta se digitar os seguintes comandos:

$ python Merge.py merge.config

$ python CISA.py cisa.config

## Ordenamento de *contigs* com rearranjos estruturais

Como dito anteriormente, é possível se utilizar diversas abordagens para se converter um conjunto de *contigs* (não ordenadas) em sequencias maiores, denominadas *scaffolds*. Além das informações provinientes do próprio sequenciamento, como leituras pareadas e leituras longas, é também possível se utilizar dados de genomas filogeneticamente próximos, denominados “referências”, para guiar este processo de construção de *scaffolds*.

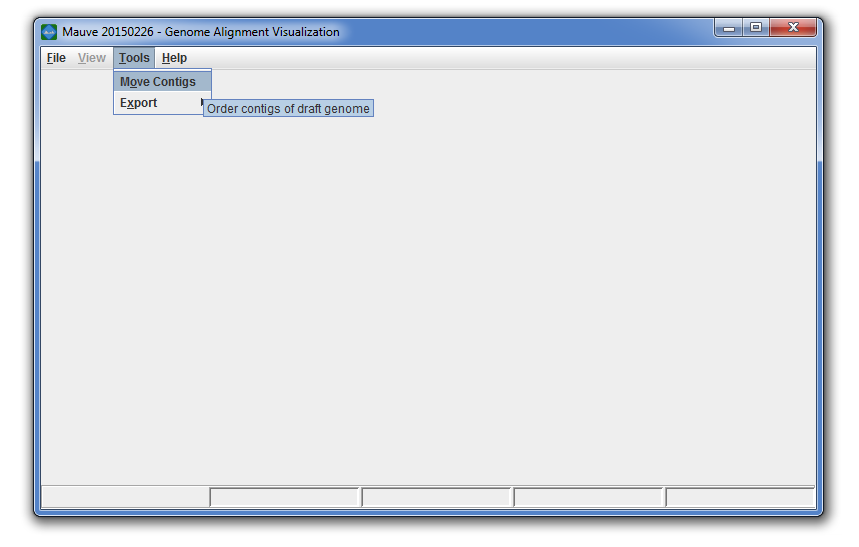
Ainda neste capítulo citamos o exemplo da ferramenta ABACAS, utiliza das informações do alinhamento das contigs contra a referência para realizar o ordenamento. Esta abordagem é computacionalmente simples, e geralmente suficiente caso os genomas sejam estruturalmente similares, entretanto, muitas bactérias apresentam elemento móveis que resultam em rearranjos estruturais observáveis mesmo em cepas muitos próximas. Desta forma, é importante que os programas que realizam o processo de ordenamento consideram a possibilidade destes eventos de forma que isso não interfiram na qualidade resultado.

### Ordenando *contigs* com Mauve

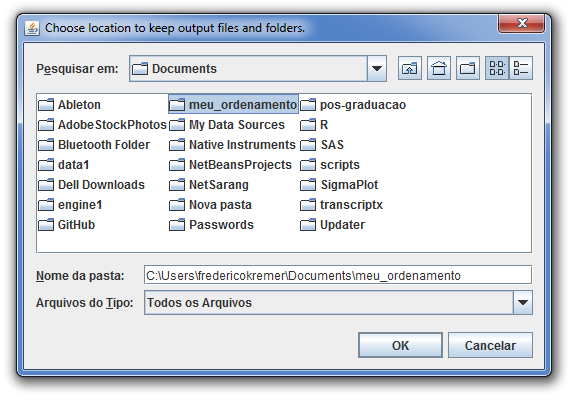
O Mauve é uma ferramenta gráfica de alinhamento desenvolvida inicialmente para a análise comparativa de genomas, mas também pode ser utilizada para o ordenamento de contigs com base em um genoma de referência. Este programa foi um dos primeiros a considerar a presenção de eventos de rearranjo estrutural durante o processo de ordenamento de contigs, o que permite a geração de *scaffolds* com maior acurácia e uma menor distorção nos resultados em decorrência do genoma de referência. O programa pode ser obtido através do endereço <http://darlinglab.org/mauve/>.

Para realizar um ordenamento de contigs utilizando o Mauve basta se abrir o programa e selecionar a opção Tools / Move Contigs.

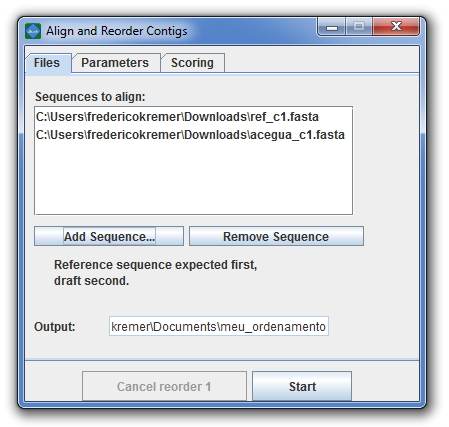
Ao selecionar esta opção será necessário indicar o diretório aonde os resultados do processo de ordenamento serão salvo.

.

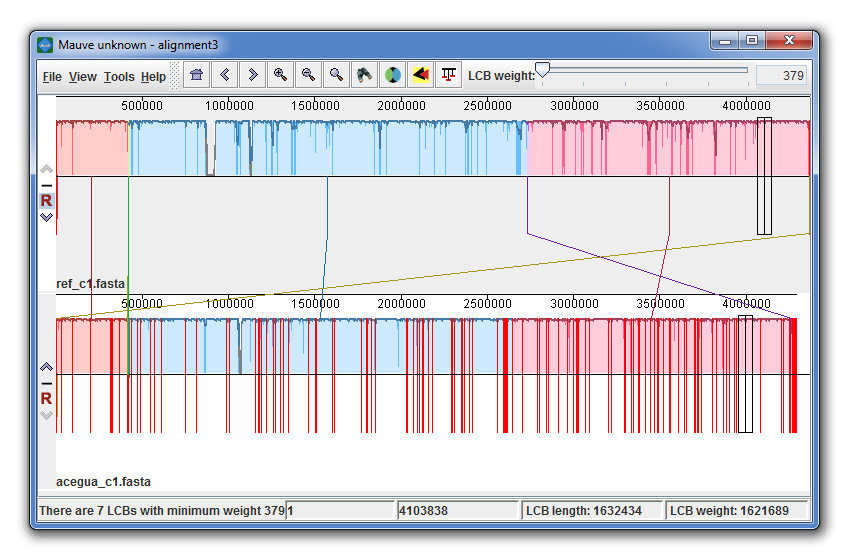
O ordenamento gerado pelo Mauve é gerado interativamente, ou seja, o processo de ordenamento é repetido com base nos resultados obtidos nos passos anteriores até que o programa chege em um resultado ótimo.



Após isso é necessário se selecionar os arquivos de entrada para o algoritmo, ambos em formato FASTA. O primeiro a ser indicado é sempre o arquivo da referência, que deve conter apenas uma sequencia, e o segundo é o do genoma a ser ordenado.



Ao final do processo o Mauve apresentará uma representação das contigs ordenadas através de um gráfico de sintenia, que indica as regiões similares entre os dois genomas.



É possível também se obter o resultado dos ordenamentos na formade um arquivo FASTA. Os arquivos FASTA de cada iteração ficam nas pastas “alignment”, dentro das pasta que foi criada pelo usuário, com o nome o mesmo nome do arquivo de *contigs* original. Para se converter este arquivo em uma scaffold é possível se utilizar um pequeno script em bash:

#!/usr/bin/env bash

#script:construir\_scaffold.sh

echo '>$1 scaffold'

while read -r line; do

if [[ $line == ">"\* ]]; then

printf "NNNNNNNNNNNNNNNNNNNNNNNN"

else

printf $line

fi

done < $1

E depois a linha de commando:

$ bash construir\_scaffold.sh contigs.fasta

## Fechamento de *gaps*

Como dito anteriormente, as *scaffolds* são compostas por *contigs* intercaladas por regiões desconhecidas, denominadas *gaps* de montagem. Estas regiões desconhecidas indicam falhas na montagem, e precisam ser removidas para uma representação do genoma a ser estudado. Para realizar este processo de remoção, também chamado “fechamento” (*gap-closing*), assim como no caso da geração de scaffolds, diferentes informação podem ser utilizadas, o que inclui leituras pareadas, leituras longas, e inclusive montagens alternativas para o mesmo genoma, de forma similar a integração de montagens.

### GMCloser

Um exemplo de ferramenta para o fechamento de gaps é o GMCloser, que pode receber diferentes arquivos auxiliares para realizar o fechamento dos *gaps*. Para rodar este programa é necessário se instalar os pacotes ncbi-blast+ e mummer, e se instalar o programa yass. O programa pode ser obtido a partir do endereço <https://sourceforge.net/projects/gmcloser/>.

$ sudo apt-get install ncbi-blast+

$ sudo apt-get install mummer

Para se realizer o fechamento dos *gaps* de um genoma com base em uma montagem alternativa baste se utilizer a seguinte linha de commando:

$ ./gmcloser --target\_scaf <scaffolds .fasta> --query\_seq <montagem alternativa .fasta> --prefix\_out <prefixo da saida>