Universitat
Autònoma
de Barcelona


# 2682-1
# Bioinformàtica::Comparació de genomes de eucariota

Memòria de Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per
Xavier Martínez Clemente
i dirigit per
Jordi Gonzàlez i Sabaté
i Mario Huerta
Bellaterra, 13 de setembre de 2011

El sotasignat, .....
Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

**CERTIFICA:**

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en

I per tal que consti firma la present.

Signat: .....

Bellaterra, .....de.....de 2011

El sotasignat, .....
Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

**CERTIFICA:**

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en

I per tal que consti firma la present.

Signat: .....

Bellaterra, .....de.....de 2011

*A la Alicia, en Jordi i l'Ivan.*

## Agraïments

La realització d'aquest treball no hauria estat possible sense el constant suport de moltes persones. Primerament, voldria agrair a tota la meva família, en especial als meus pares, per l'educació que m'han sabut impartir, per totes les oportunitats que m'han donat i per haver estat sempre al meu costat de forma incondicional. Al meu germà, per fer-me obrir els ulls en més d'una ocasió i per tot el que he après gracies a ell. A en Mario Huerta, per la seva predisposició a ajudar en tot moment, i el seu entusiasme contagiós que em va donar les forces que necessitava en alguns moments. A en Jaume Moreno, i la seva col·laboració fins i tot, amb la recent operació de la vista. Als meus amics, amigues, i tots els que m'han acompanyat durant aquesta etapa i l'han fet única i irrepetible.

## ÍNDEX DE CONTINGUTS

# 1. Introducció

## 1.1 Motivació

El desconeixement i la curiositat per la genòmica ja hi eren, la oportunitat de fer-los desaparèixer va ser un dels principals motius pels que em vaig decantar per la servidorrealització d'un treball d'aquest tipus. El fet de pensar que tots nosaltres, el complexe funcionament del nostre cos, el nostre instint, les nostres malalties..., estan codificades en una petita regió d'una cèl·lula qualsevol, resulta prou interessant com per endinsar-se a qualsevol projecte on es pugui aprendre i col·laborar amb aquest interessant món.

A més, aquest treball em resultava un d'ambiciós, actual, punter, de gran utilitat i repercussió internacional i tot plegat em presentava un repte personal que volia superar. Tot l'esforç invertit seria d'utilitat per tothom que volgués estudiar i obtenir resultats de les comparacions de genomes eucariotes, com l'estudi de gens que provoquen càncer o altres patologies i d'aquesta manera faria una aportació a la medicina.

A l'aspecte tècnic em vaig trobar amb un problema a solucionar d'una dificultat considerable, on s'havien d'aplicar totes les eines d'un enginyer, si volíem obtenir els resultats esperats. S'havia de fer ús d'un potent servidor (i aprofitar-lo al màxim) per resoldre els costosos càlculs que van darrera d'una comparació de genomes. Havia de treballar amb GNU/Linux, shell-scripting i C++ per realitzar una automatització total del servidor, realitzar la paral·lelització de codi i pensar nous algorismes per solucionar el problema. Tot això em va donar l'empenta final per decidir-me per aquest treball, doncs implicava l'ús de tecnologies potents, amb un grau de llibertat alt, molt esteses i on desitjava profundament actualitzar i ampliar al màxim els meus coneixements.

## 1.2 Estat del art

Un genoma és la totalitat d'informació genètica que conté un organisme viu. Aquest genoma està composat per una llarga seqüència de les quatre bases nitrogenades: A,C,T i G.

Si realitzem la comparació de genomes, podem obtenir una gran informació dels processos evolutius que han portat a l'existència de la gran varietat d'éssers vius que avui poblen la Terra. A més, aquesta comparació de genomes, té una important aplicació mèdica. De fet, l'assignació de funcionalitat als gens o per exemple, la detecció de gens que poden provocar malalties, s'acostuma a fer pel reconeixement de seqüències funcionals que es conserven d'un ésser viu a un altre.

Els genomes es classifiquen en tres tipus: Arqueobacteris, Bacteris i Eucariotes. Els genomes d'eucariota són enormes, a diferència dels genomes de bacteris i arqueobacteris, i això és un problema per la comparació [Figura 1]. Per exemple, el genoma humà té tres mil milions de bases mentre que el genoma d'un bacteri té al voltant de tres milions, ens podem fer una idea.

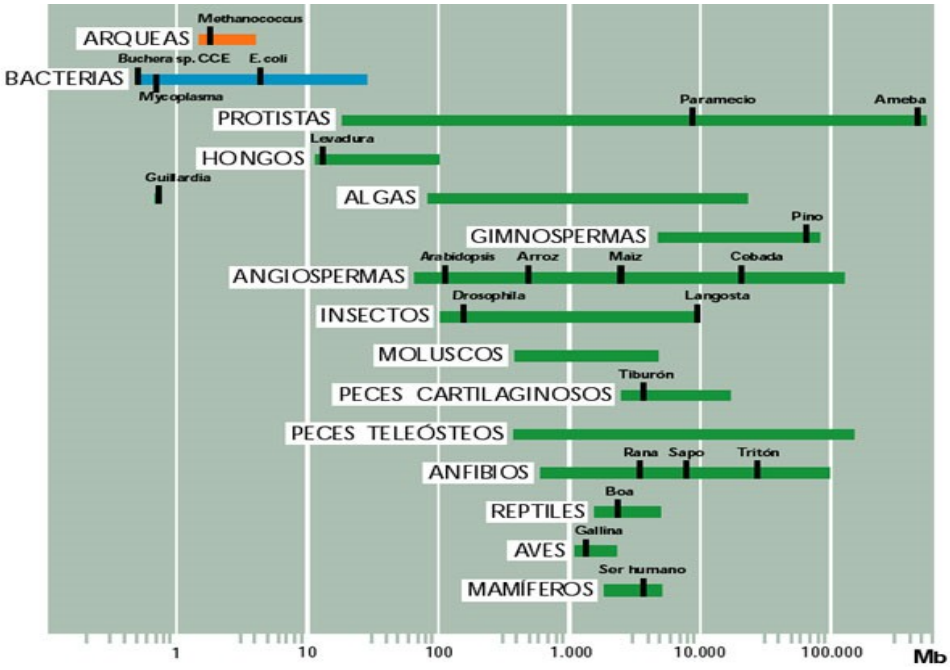



Figura 1: A la imatge podem apreciar les diferencies de mides de genomes mesurat en milions de bases, per a cada gran grup de genomes. En color taronja els arqueobacteris, en color blau els bacteris i en verd les eucariotes.

Una altre diferència bàsica dels genomes eucariota amb la resta, es que les eucariotes inclouen introns. Els introns són seqüències de codi genètic de control no codificant i que no s'arriben a expressar, és a dir, la seva seqüència no s'utilitza en el moment de sintetitzar la proteïna.

A més els genomes eucariota estan formats per diversos cromosomes i tots junts formen el genoma complet. Els bacteris en canvi, només tenen un cromosoma.

Amb aquest gran volum de dades complexes si es vol automatitzar el procés de comparació de genomes, s'ha de realitzar un procediment diferent per a comparar bacteris o arqueobacteris i per a comparar eucariotes.

Existeixen diferents formes de comparar genomes. La tècnica que es fa servir a la línia d'investigació del IBB per a la comparació de genomes complets es basa en trobar les seqüències comuns úniques més grans entre els genomes, anomenades MUMs (Maximal Unique Matching) [5].

L'algoritme que calcula els MUMs es diu MUMOL [5][8], desenvolupat per Mario Huerta (co-director del projecte) i Xavier Messeguer que van aconseguir un mètode eficient en temps/espai pel calcul dels MUMs.

Els MUMs són les sub-seqüències de mida més gran i úniques de bases nitrogenades comuns als dos genomes que comparem. La totalitat de MUMs trobats formen el que s'anomena "skeleton" sobre el que es fa la comparació global.

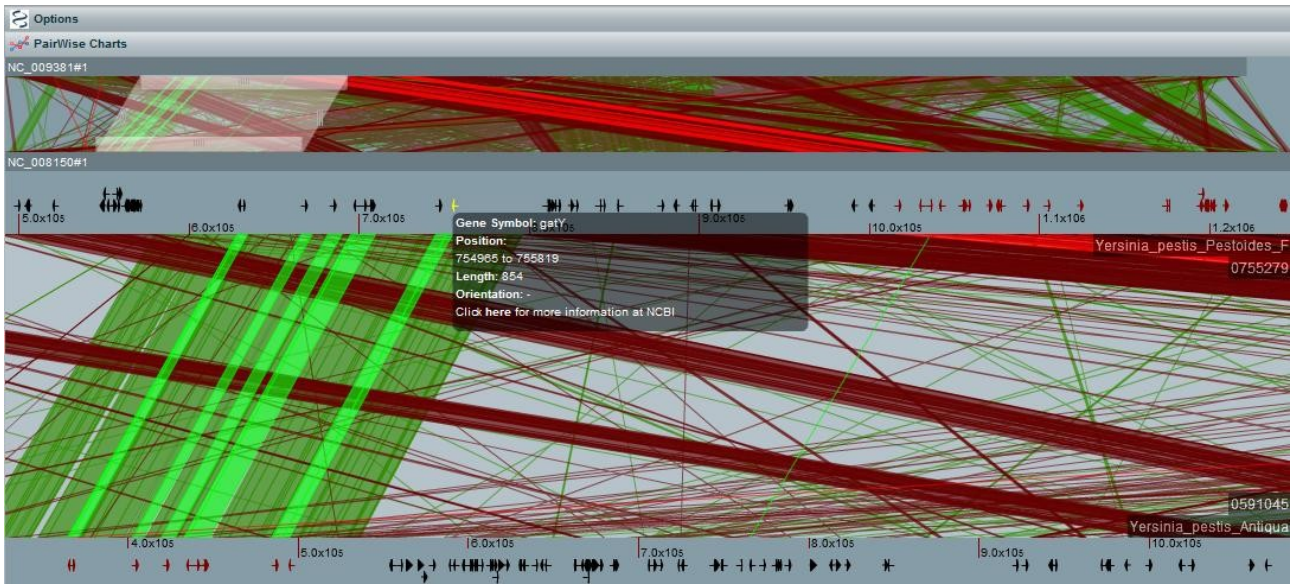
Exemple de sub-seqüències de MUMs:
agctcgat**GGGCTTTAGACTCTCGATA**ggcgcagag**GCTCGCTAGAATCGCTAGATC**ac
agacctaa**GGGCTTTAGACTCTCGATA**agtctatcc**GCTCGCTAGAATCGCTAGATC**ta

El concepte de SuperMUMs va sorgir amb la idea de fer viable la comparació de genomes grans com els d'eucariotes. Un SuperMUM o SMUM és una agrupació de MUMs consecutius que ha de verificar els següents requisits:

- L'ordre dels MUMs que formen el SMUM al primer genoma, ha de ser el mateix que al segon genoma.
- L'espai (gap) entre un MUM i el següent no pot ser més gran que la longitud d'aquests sumats multiplicats pel multiplicador de gap.
- Un SMUM no pot estar inclòs dins d'un altre SMUM.
- Els MUMS que estan inclosos dins d'un SMUM queden absorbits per aquest.

Si ens parem a pensar, quan passem d'operar amb MUMs, on treballem amb cadenes de coincidència exacte, a un SMUM, on treballem amb cadenes de coincidència aproximada, estem generant això, aproximacions. Però, gracies a aquestes aproximacions dels SMUMs, podem trobar grans regions del genoma amb petites diferencies evolutives dins d'ella. A més, estaríem ignorant les diferencies evolutives dels introns, que com em explicat abans, són regions no codificants i que no sintetitzen la proteïna. Per tant, els gens de ambdós genomes estarien sintetitzant la mateixa proteïna encara que una petita part del codi genètic hagués canviat.

De vegades ens podem trobar que els MUMs siguin inversos, això vol dir que una base de la sub-seqüència comuna a la posició *N* a un genoma és igual a la base de la sub-seqüència a l'altre genoma a la posició *N* però començant pel final d'aquesta.
Això ens indica que hi ha hagut una inversió evolutiva del codi genètic, és a dir que la seqüència comuna s'ha girat.




[Figura 2]: Interfície gràfica via web al servidor per a comparació de genomes del IBB-UAB: http://platypus.uab.cat [4] que mostra MUMs i SMUMs directes (línies verdes) i inversos (línies vermelles) així com els gens dels genomes comparats. A la figura es mostra la comparació entre els Bacteris Yersinia_pestis_Pestoides_F i Yersinia_pestis_Antiqua.

La recerca clàssica de MUMs es realitza mitjançant la construcció de Suffix-tree's, representant tots els sufixos de les seqüències en un arbre que comparteix els seus prefixos comuns (prefixos dels sufixos). El procés de construcció de l'arbre té un alt cost en temps de procés. La variant utilitzada pel algorisme MUMOL[5][8] permet buscar els MUMs de diverses seqüències, construint l'arbre per una seqüència i recorrent-lo després per comparar amb la resta. Aquestes característiques suposen un gran estalvi de temps i espai en comparació amb altres programes existents, ja que el temps és lineal respecte a la longitud de tots els genomes i l'espai utilitzat és lineal respecte a la longitud del genoma més petit [5].

Existeixen aplicacions que fan ús del MUMOL com el MALGEN (2003) [6] eina web amb l'acrònim de Multiple ALignment of GENomes de bacteris. És una eina web per la exploració de relacions entre seqüències de ADN. Una altre aplicació seria el M-GCAT (2006) [7] acrònim de Multiple Genome Comparison and Alignment Tool i és una eina interactiva d'escriptori per la comparació de diversos genomes simultàniament.

Els genomes de bacteris i eucariotes estan disponibles al servidor públic del NCBI (National Center for Biotechnology Information) [1]. En aquest servidor també es disposa de tota la informació referent a cada genoma disponible. I d'aquest mateix servidor (NCBI) podem obtenir els gens de cada genoma amb les posicions que ocupen dins del genoma.

Com que la genòmica evoluciona cada dia i la majoria de genomes encara no estan complets, la actualització del servidor és bastant freqüent. Cada fitxer està etiquetat amb la seva data de modificació pel que podem saber quan un genoma s'ha actualitzat. Els genomes que tenen una seqüenciació bastant avançada, trobem que tenen diferents versions. A mesura que es van identificant nous gens a un genoma també es van actualitzant els fitxers amb els mapes de gens sobre el genoma.

L'institut de Biotecnologia i de Biomedicina (IBB) [9] segueix una línia d'investigació per a la comparació de genomes sencers. Per això, està desenvolupant un servidor públic per la consulta web de comparacions de genomes, http://platypus.uab.cat [4]. Existeix tot un procés de comparació i visualització de bacteris i arqueobacteris que comentarem breument a continuació.

- Implementació del MUMOL per la generació de MUMs donats dos genomes.
- Calcul de SMUMs per trobar SMUMs a partir de MUMs.
- Programa en Java pel tractament dels fitxers que contenen el mapatge dels gens del genoma. Funcional en Homo Sapiens i Macaca Mulatta.
- Mummy és una interfície web desenvolupada en FLEX per la visualització de comparacions de genomes. És disponible al nostre servidor, platypus.uab.cat [1]. Es va aconseguir un sistema per l'exploració de les comparacions de genomes de forma fluida i per grans volums de dades. S'han realitzat exitoses proves amb els genomes de Homo Sapiens i Macaca Mulatta.

Degut a les diferències amb els genomes de bacteris i arqueobacteris vistes anteriorment, automatitzar la comparació d'eucariotes no és una feina trivial:

-La descarrega de eucariotes del servidor del NCBI [1] no és fàcil d'automatitzar. Cada espècie eucariota té la seva carpeta, i sense cap tipus de classificació ni organització: genomes vàlids, genomes sense seqüenciar, mamífers, plantes, tots barrejats.

-La mida dels genomes d'eucariotes fa quasi impossible la tasca de comparació en aquests genomes. Suposant una mida mitjana de genoma d'eucariota de 1,5Gb, per cada genoma que vulguem comparar i suposant que és el genoma petit dels dos genomes comparats, necessitaríem un total aproximat de 165Gb de memòria RAM per realitzar la comparació.

-Aquesta gran mida també ens fa preveure que l'actual tractament per trobar SMUMs serà insuficient i que augmentar la mida del multiplicador de gap arbitràriament no és la solució, dons ens hem d'assegurar que realment s'estan fusionant en un supermum, mums suficientment grans i que formen part del "skeleton" . Donat que les regions no codificants com els introns poden ser diferents en un genoma i un altre sense variar el gen, aquesta millora en el tractament dels SMUMs es una prioritat.

## 1.3 Objectiu/s del projecte

Els genomes dels éssers vius contenen la informació genètica de cada espècie. Volem realitzar comparacions de genomes d'eucariota de forma automàtica a mesura que aquests genomes vagin essent seqüenciats. Volem que tota la informació de cada genoma estigui contínuament actualitzada, i que els resultats de les comparacions siguin accessibles via web.

Els genomes d'eucariota són els més grans, i els volem comparar de forma eficient, de forma que el servidor pugui mantenir el ritme d'aparició de noves especies amb els seus genomes seqüenciats i incorporar aquest genomes al conjunt de comparacions entre genomes. Tot de forma que els resultats serveixin tant per estudis biomèdics com per estudis evolutius d'espècies, que es realitzaran mitjançant una aplicació web al servidor http://platypus.uab.cat [4].

Els objectius a assolir per aconseguir la meva fita serien aquests:

- Automatització completa dels càlculs per generar els fitxers resultat que necessita l'interfície web sobre la comparació dels genomes d'eucariota.

- Modificar la comparació de genomes per bacteris i arqueobacteris adaptant-la a eucariotes. Especialment, la generació de MUMs. L'algoritme actual està preparat per la comparació de dos genomes complets. Per problemes de memòria, en genomes d'eucariotes, no podem comparar dos genomes complets, pel que hem de comparar els diferents cromosomes d'un genoma contra l'altre genoma. Això és perquè la mida dels genomes eucariotes és tan gran que necessitaríem més de 300Gb de RAM per poder realitzar tranquil·lament la majoria de comparacions, genoma contra genoma complet, cosa bastant inviable. Aquest fet ens obliga a realitzar la comparació per cromosomes.
  - Al comparar els cromosomes per separat contra un mateix genoma, es poden produir MUMs que no són únics perquè apareixen a més d'un cromosoma, i els hem d'eliminar. A més hem de unir els MUMs generats per cada cromosoma, de forma ordenada i transformant les posicions dins del cromosoma per posicions dins del genoma.

- Donat que la comparació de dos genomes d'eucariota trigaria entre 1 i 2 dies, paral·lelitzaré els calcul de la comparació. El procés de paral·lelització serà totalment automàtic i autònom, sempre llançant les comparacions més adients en cada moment, si hi ha recursos per a una comparació més gran, dons una més gran, sinó llençarà de més petites, però sempre utilitzant al màxim els recursos de la màquina. D'aquesta forma es podrà mantindre el ritme de seqüenciació de noves especies, aprofitant l'amplia memòria RAM del servidor http://platypus.uab.cat [4], un total de 63Gb i la seva arquitectura multicore amb 24 processadors.

- Adaptar el procés de generació de SMUMs a eucariotes. Al tractar genomes tant grans, no tenim prou amb una sola execució del programa de generar SMUMs. Realitzant varies passades al programa de calcul de SMUMs, farem que els SMUMs resultants no siguin només agrupacions de MUMs separats per gaps, sinó agrupacions de SMUMs separats per gaps. D'aquesta forma no fa falta permetre un gap mes gran, sinó que el que augmenta és la mida dels elements que es fusionen en el nou SMUM. Que ja no seran MUMs sinó SMUMs obtinguts en anteriors passades.
  - Permetre l'execució del programa de càlcul de SMUMs amb entrada de fitxers SMUMs en comptes de MUMs. Trobar el multiplicador més adient per la generació de SMUMs i el nombre adient de passades.

- Automatització del programa encarregat de generar el mapatge de gens sobre el genoma. Els fitxers de mapatge de gens permeten visualitzar quins gens s'estan conservant d'un organisme a un altre a l'interfície gràfica web Mummy. Per això, el programa ha de descarregar els fitxers amb la informació genètica de cada cromosoma i situar cada gen a la seva posició corresponent dins el genoma.

- Actualització automàtica bimensual de les comparacions entre genomes amb nous genomes seqüenciats o canvis significatius en la versió d'un genoma al repositori mundial de genomes del NCBI (National Center for Biotechnology Information) (FTP) [2]. La majoria de genomes d'eucariota no estan completament seqüenciats, pel que utilitzen un sistema de versions per controlar els canvis.
  - Si els canvis a la seqüenciació d'un genoma són importants, el robot d'actualització haurà de baixar el nou genoma i realitzar la comparació de nou. A més, s'hauran de detectar canvis de posició de gens, així com el descobriment de nous gens fins ara desconeguts per a cada especie en qüestió.
  - Per cada nou genoma trobat al servidor del NCBI i encara no incorporat al nostre servidor, el robot d'actualització haurà d'esbrinar si es tracta d'un genoma d'eucariota i la seva classificació dins dels diferents tipus de eucariotes (pe. esbrinar si es tracta d'un animal, planta, fong, i les seves subclasses). Per aquesta tasca es farà ús de les eines de consulta remota a les bases de dades del NCBI, les e-utils [3]. D'aquesta manera podrem classificar l'especie descarregada al nostre servidor [4].

## 1.4 Organització de la memòria

A continuació, exposaré el plantejament seguit per reproduir al present document tot el treball realitzat.

Al següent apartat faré una revisió dels fonaments teòrics que engloben el marc de treball. La intenció és explicar tots els conceptes bàsics per permetre al lector estar situat en tot moment.

Després dels fonaments teòrics, passaré a explicar totes les fases seguides per arribar als objectius marcats. En la secció fases explicaré la forma en que hem aconseguit els objectius i com s'han resolt cadascun dels problemes sorgits.

-En primer lloc, explicaré la fase inicial: adquirir els coneixements bàsics, reproduir les proves de comparacions fetes de homo sapiens amb macaca mulatta i finalment provar de comparar altres espècies.

-A la fase següent es va desenvolupar l'automatització de la generació de MUMs en eucariotes, passant a comparar per cromosomes en comptes dels dos genomes sencers.

-Un cop tenim un sistema de càlcul de MUMs automatitzat per eucariotes, a la següent fase desenvolupo la paral·lelització per obtenir temps de execució acceptables.

-A la quarta fase es desenvolupa el càlcul de SMUMs a eucariotes.

-A la cinquena fase es desenvolupa l'automatització del programa de mapatge de gens. Veurem tota la problemàtica sorgida i les solucions proposades per aconseguir resoldre-la.

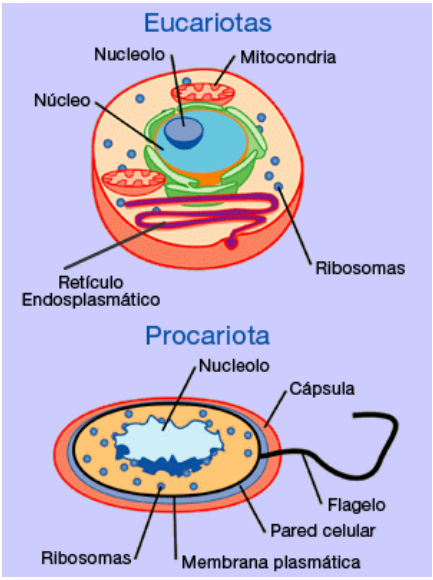
-L'última fase correspon a l'automatització del robot de descarrega de nous genomes eucariotes i de comparació dels genomes descarregats.

Per finalitzar, a la secció de informe tècnic, tractarem de plasmar tot el treball realitzat per permetre el reutilitzament, adaptació i millora del software desenvolupat. Es tractarà al nivell necessari perquè futurs desenvolupadors puguin entendre i continuar amb la línia de treball.

# 2. Fonaments teòrics

## 2.1 Conceptes biològics bàsics

Una cèl·lula és l'element més petit que es pot considerar viu. Depenent de les característiques de la cèl·lula d'un organisme, els podem classificar. Per posar un exemple, si la cèl·lula disposa d'un nucli al medi intracel·lular, podem determinar que la cèl·lula pertany a un organisme de eucariota [Figura 3], les cèl·lules del ésser humà.




[Figura 3] Diferències entre els tipus de cèl·lula eucariota i procariota.

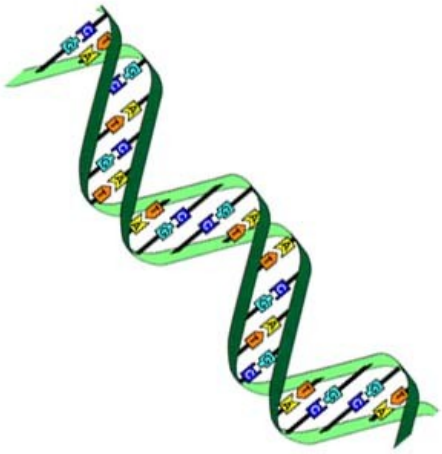
Les principals molècules que podem trobar a una cèl·lula són els glúcids, lípids, proteïnes i àcids nucleics.

-Els glúcids i lípids tenen funcions tant estructurals com energètiques. Entre els glúcids més típics trobem la glucosa o la fructosa i entre els lípids podem trobar els Omega-3 o el colesterol.

-Les proteïnes són una successió d'aminoàcids constituïda sobre un alfabet de 20 símbols on cada símbol és un dels aminoàcids possibles. Les proteïnes regulen el metabolisme de la cèl·lula i tenen funcions estructurals, motrius, catalítiques, o fins i tot sensorials. Depenent del conjunt de proteïnes que regulin el metabolisme de la cèl·lula, es determinarà l'especialització de la cèl·lula i el seu funcionament.

-Els àcids nucleics, més coneguts per ADN, àcid desoxiribonucleic, i ARN, àcid ribonucleic, composen una successió de nucleòtids que es poden distingir per la base nitrogenada que els constitueix. Els quatre tipus de bases nitrogenades a l'ADN són A (adenina), T (timina) , G (guanina) i C (citosina). Al ARN podem trobar les anteriors, però substituint la T (timina) per U (uracilo).

L'ADN està format per dos cadenes helicoïdals unides per les bases nitrogenades [Figura 4] que hem esmentat abans però en relacions complementaries. A sempre va amb T i G sempre va amb C. La funcionalitat del ADN és transmetre la seqüència de nucleòtids que la composa i la del ARN comporta funcions variades com catalítiques o de regulació de síntesi proteica.



[Figura 4] Cadenes helicoïdals unides per les bases nitrogenades complementades.

Tota la informació que es codifica a la seqüència de les bases nitrogenades conté les possibles proteïnes que es generaran i els ARN que regularan el funcionament de la cèl·lula, entre d'altres. Aquestes sub-seqüències del ADN les anomenem gens.

D'ARN existeixen diferents tipus i s'acostume a anomenar amb un sufix de la funció que realitzen. ARNm dins de la molècula de ARN  fa de missatger, transportant la informació d'un gen al Ribosoma (orgànul de la cèl·lula encarregat de generar les noves proteïnes).

Resumint, el codi original del ADN està format per quatre símbols (A,T,G,C) i es tradueix a una proteïna formada per una seqüència codificada amb 20 símbols. Així, les agrupacions de tres nucleòtids ens permeten codificar un dels vint aminoàcids que sintetitzarà amb certa redundància. Aquesta redundància es pot interpretar com rastres evolutius o tolerància a errors.

El fet que un gen s'expressi o no, és a dir, que generi la proteïna o ARN que codifica, depèn de diversos factors, com pot ser l'existència de molècules al medi cel·lular que bloquegin la seva lectura. La aparició d'aquestes molècules ve donada per l'expressió d'altres gens, així que ens podem fer una idea de les complexes dinàmiques que regulen el control de una cèl·lula.

Un genoma és la totalitat d'informació genètica que conté un organisme codificat a les seves molècules d'ADN. Podem adonar-nos que l'estudi dels gens del genoma d'una espècie pot aportar-nos multitud d'informació referent al metabolisme de la cèl·lula i per tant informació del organisme en qüestió. Fins i tot, les parts no codificants del genoma, com els gens que comentàvem abans que no s'expressen, es creu que participen activament a la dinàmica cel·lular.

## 2.2 La genètica

La genètica és l'estudi de la natura, organització, funció, expressió, transmissió i evolució de la informació genètica codificada als organismes. Podem distingir les següents àrees:

-Genètica clàssica: Estudi de la transmissió i localització de gens als cromosomes.

-Genètica molecular: Estudi de l'estructura i control d'expressions genètiques.

-Genètica evolutiva: Estudi de processos evolutius d'espècies.

-Genòmica: Anàlisis i interpretació dels genomes.

El treball actual s'emmarca dins de la genòmica comparativa, una de les àrees més avantguardistes de la biologia, i estudia les relacions entre els genomes de diferents espècies amb l'objectiu d'entendre el funcionament dels organismes i obtenir la informació que proporciona les diferències evolutives a partir de la selecció natural.

## 2.3 L'evolució

L'evolució biològica és el conjunt de transformacions o canvis a través del temps que origina la diversitat de formes de vida que poblen avui la terra a partir de un avantpassat comú. Els diferents processos que afecten a aquestes transformacions són:

-Selecció natural: Adaptació al medi i al entorn.

-Deriva genètica: Dinàmiques dins d'una població.

-Mutació: Variacions espontànies dins del codi genètic.

-Flux genètic: Migració entre poblacions.

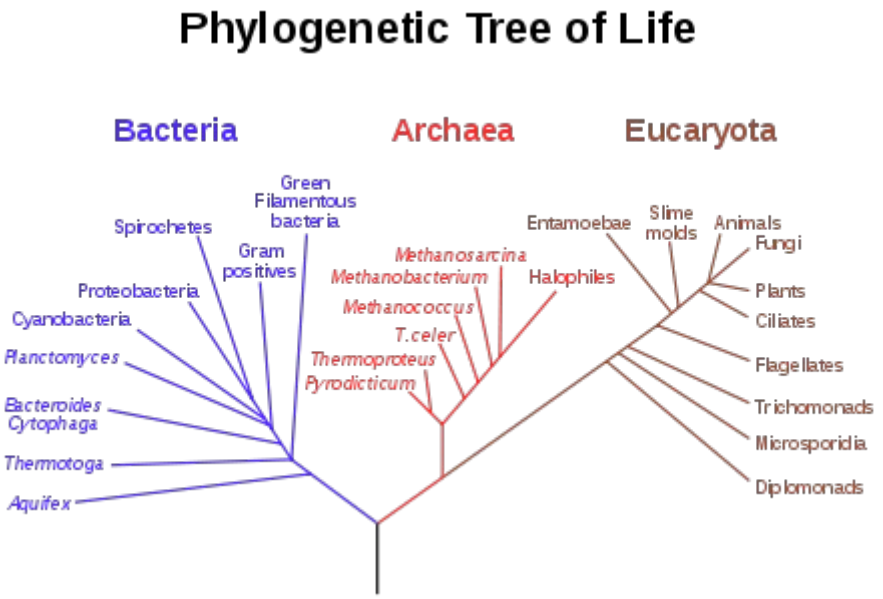
Considerant aquests conceptes, la comparació genòmica de les espècies ens servirà per veure la proximitat evolutiva, la detecció de característiques que han perdurat a les espècies i veure els gens que han perdurat.

Per cadascuna de les espècies que existeixen avui en dia, podríem fer un estudi de quina és la espècie de la que han evolucionat i trobar així els ancestres comuns. La cerca de ancestres comuns és el mateix que cercar regions de genoma que es conserven a un altre. Si trobem una regió que es conserva, vol dir o bé que un genoma es ancestre de l'altre o que ambdós son descendents d'un ancestre comú.

Aquesta cerca de similituds entre regions de diferents genomes aporta molta informació sobre les relacions de les espècies i com les propietats d'un gen són assignades a un altre gen.

## 2.4 L'arbre filogenètic

Les espècies poden estar classificades en un arbre filogenètic [Figura 5], això vol dir que les espècies es classifiquen científicament segons les seves relacions de proximitat amb altres espècies. D'aquesta manera estem reconstruint la història de la diversificació o filogènesis de les espècies des de l'origen de la vida fins avui dia.




[Figura 5] Podem observar un exemple dels tres grans grups de espècies classificades en un arbre filogenètic.

Veiem que els tres grans dominis a la classificació de cèl·lules (i segons aquestes, la classificació d'espècies també) són els arqueobacteris, bacteris i eucariotes [Figura 6]. Els arqueobacteris i bacteris pertanyen al grup de les cèl·lules sense nucli diferenciat, les procariotes, que tenen l'ADN dispers al citoplasma. Les eucariotes són més complexes i les seves cèl·lules contenen un nucli cel·lular que emmagatzema l'ADN. A més, tenen orgànuls com els mitocondris, que regulen la síntesis del combustible del metabolisme cel·lular (l'ATP), que contenen ADN propi i es creu que provenen de simbiosis anteriors amb altres procariotes

| Tipo | Reino | Descripción |
|---|---|---|
| Bacteria (Eubacteria) | mycoplasmas<br>cyanobacteria<br>bacterias Gram-positivas<br>bacterias Gram-negativas | ▪ Son procariotas, contiene el ADN en el citoplasma.<br>▪ Son organismos microscópicos y en su mayor parte unicelulares.<br>▪ Son sensibles a los antibióticos antibacterianos tradicionales. |
| Archaea (Archaebacteria) | metanógeno<br>halófilos extremos<br>termoacidófilos | ▪ Son procariotas, contiene el ADN en el citoplasma.<br>▪ Son organismos unicelulares.<br>▪ No son sensibles a algunos antibióticos que afectan a las Bacterias.<br>▪ Viven a menudo en ambientes extremos. |
| Eucariota (Eukaryota) | Hongos<br>Protistas<br>Plantas<br>Animales | ▪ Tienen un núcleo celular diferenciable que contiene el ADN.<br>▪ No son sensibles a los antibióticos antibacterianos tradicionales. |

[Figura 6] Veiem les característiques de cadascun dels tres grans grups d'espècies.

## 2.5 La Bioinformàtica

La bioinformàtica és la aplicació de la tecnologia computacional a les mans dels problemes biològics. És un camp d'estudi multi-disciplinari que necessita de l'aplicació de la informàtica, les matemàtiques, la estadística, la intel·ligència artificial, la química i la bioquímica. Els volums d'informació que pot tenir un genoma eucariota, implica la necessitat d'automatitzar els processos cerca de estructures i patrons als genomes i de comparacions de seqüències. Al nostre treball les estructures de dades que traiem de les comparacions de genomes i que haurem de tractar són els MUMs.

## 2.6 Les estructures de MUMs

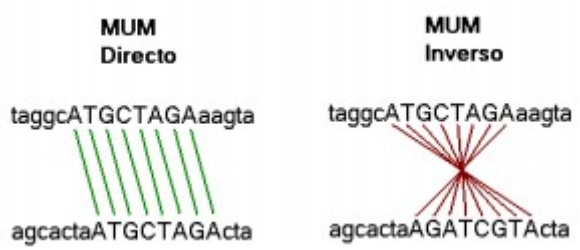
Un MUM o maximal unique matching, és la subseqüència única màxima que coincideix a la comparació de dos genomes. Dit d'altre forma, es un fragment de codi genètic on les bases nitrogenades coincideixen correlativament amb les d'un altre genoma, aquesta coincidència es la més llarga i no es repeteix.

A més d'agrupar la informació útil en un format més fàcilment tractable que les bases nitrogenades, el conjunt de MUMs formen l'esquelet (skeleton) de la comparació de genomes, la seva estructura bàsica que té un significat biològic. És lògic dons, que a les comparacions de genomes de espècies evolutivament properes, trobem gran quantitat de MUMs, o que els que hi hagin  tinguin una mida gran.

Si trobem MUMs de gran llargària vol dir que els fragments d'ADN han perdurat intactes a les dos espècies i que la funció dels gens que codifiquen haurien de ser les mateixes (tot i que hem explicat abans que la expressió d'un gen depèn de molts altres factors). També hem dit que els MUMs han de ser únics, això és per descartar fragments del codi genètic que es repeteixen i no ens aporten cap informació biològica.

A les subseqüències dels genomes, per exemple el codi que representa un gen, no té pre-establert cap ordre de lectura. Això vol dir que els gens poden estar codificats de esquerra a dreta o de dreta a esquerra respecte la seqüència del genoma. Per tant, podem trobar fragments de codi, que durant el procés evolutiu inverteixin el seu ordre d'una especie a una altre i mantenint la mateixa funcionalitat.

Els MUMs contemplen aquest fet, i a la cerca de MUMs els classifiquem en MUMs directes si han mantingut l'ordre o en MUMs inversos si l'han invertit [Figura 7].




[Figura 7] Veiem dos exemples, en verd un MUM directe i en vermell un MUM invers.

## 2.7 Les estructures de SMUMs

Per a genomes grans com els de les eucariotes, és imprescindible agrupar aquests MUMs obtinguts si volem poder tractar la informació amb facilitat. La idea és que podríem trobar seqüències coincidents encara més llargues de no ser per petites bases que s'han alterat i que impedeixin la formació de MUMs grans. Amb el concepte de SMUM, tenim dos avantatges, un és poder fer viable la comparació de genomes enormes com els d'eucariotes, i l'altre és aplicar una mica de tolerància a la cerca de seqüències coincidents, que poden haver variat mínimament.

Per crear SMUMs a partir de MUMs s'han de complir una serie de condicions:

-L'ordre dels MUMs que formen el SMUM al primer genoma, ha de ser el mateix que al segon genoma.
-L'espai (gap) entre un MUM i el següent no pot ser més gran que la longitud d'aquests sumats multiplicats pel multiplicador de gap.
-Un SMUM no pot estar inclòs dins d'un altre SMUM.
-Els MUMS que estan inclosos dins d'un SMUM queden absorbits per aquest.

Podem detectar que, per definició, un SMUM pot originar superposicions, mentre que els MUMs mai es superposen

# 3. Fases

## 3.1 Planificació

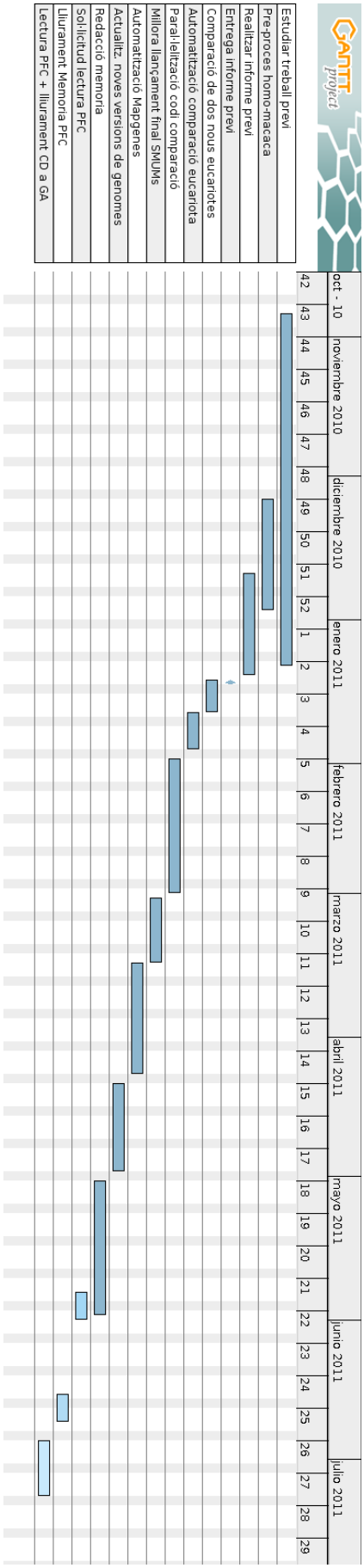
En aquest apartat, podrem comprovar les diferències entre la planificació inicial i la planificació real. Explicaré els motius de la variació de la planificació resultant del informe previ i la planificació final un cop hem acabat el treball.

Veiem seguidament la planificació inicial prevista pel desenvolupament del treball [Figura 8]. Podem observar les distintes etapes planificades i les dates que pensàvem que trigaríem. A la pàgina següent posaré el diagrama de Gantt corresponent [Figura 9].

| Nombre | Fecha de ini... | Fecha de fin |
|---|---|---|
| Estudiar treball previ | 27/10/10 | 11/01/11 |
| Pre-proces homo-macaca | 6/12/10 | 30/12/10 |
| Realitzar informe previ | 22/12/10 | 13/01/11 |
| Entrega informe previ | 14/01/11 | 15/01/11 |
| Comparació de dos nous eucariotes | 14/01/11 | 21/01/11 |
| Automatització comparació eucariota | 21/01/11 | 29/01/11 |
| Paral·lelització codi comparació | 31/01/11 | 1/03/11 |
| Millora llançament final SMUMs | 2/03/11 | 16/03/11 |
| Automatització Mapgenes | 16/03/11 | 9/04/11 |
| Actualitz. noves versions de genomes | 11/04/11 | 30/04/11 |
| Redacció memoria | 2/05/11 | 31/05/11 |
| Sol·licitud lectura PFC | 26/05/11 | 1/06/11 |
| Lliurament Memoria PFC | 17/06/11 | 23/06/11 |
| Lectura PFC + lliurament CD a GA | 27/06/11 | 9/07/11 |

[Figura 8] Planificació inicial de etapes i dates. Previsió abans del treball

La planificació inicial va resultar complexa a l'hora de preveure dates. Treballem amb dades enormes i del món real. El no tenir un conjunt de dades delimitat, sinó que cada dia va augmentant i variant, provoca problemes inesperats que fan variar la planificació.




[Figura 9] Diagrama de Gantt de la planificació inicial. Previsió abans del treball

Ara observarem la planificació real resultant del desenvolupament de l'automatització [Figura 10]. De la mateixa forma que em fet abans, posaré a la següent pàgina el diagrama de Gantt final [Figura 11].

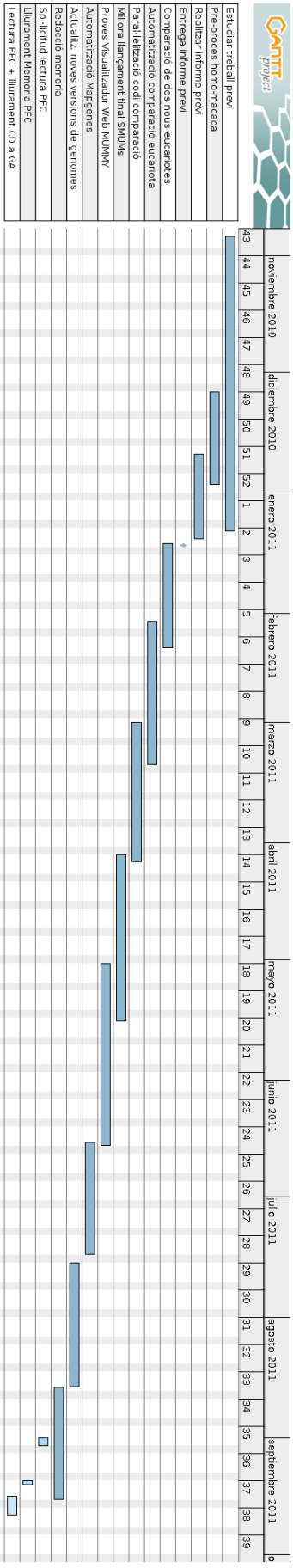
| Nombre | Fecha de inicio | Fecha de fin |
|---|---|---|
| Estudiar treball previ | 27/10/10 | 11/01/11 |
| Pre-proces homo-macaca | 6/12/10 | 30/12/10 |
| Realitzar informe previ | 22/12/10 | 13/01/11 |
| Entrega informe previ | 14/01/11 | 15/01/11 |
| Comparació de dos nous eucariotes | 14/01/11 | 10/02/11 |
| Automatització comparació eucariota | 3/02/11 | 12/03/11 |
| Paral·lelització codi comparació | 1/03/11 | 6/04/11 |
| Millora llançament final SMUMs | 4/04/11 | 17/05/11 |
| Proves Visualitzador Web MUMMY | 2/05/11 | 18/06/11 |
| Automatització Mapgenes | 17/06/11 | 16/07/11 |
| Actualitz. noves versions de genomes | 18/07/11 | 19/08/11 |
| Redacció memoria | 19/08/11 | 17/09/11 |
| Sol·licitud lectura PFC | 1/09/11 | 5/09/11 |
| Lliurament Memoria PFC | 12/09/11 | 13/09/11 |
| Lectura PFC + lliurament CD a GA | 16/09/11 | 21/09/11 |

[Figura 10] Planificació real de etapes i dates.

Podem observar com a la totalitat del treball s'ha allargat tres mesos respecte a la planificació inicial. La explicació és senzilla, originalment aquest treball estava plantejat per ésser realitzat en dos semestres. Donat que al primer semestre estava treballant, no disposava del temps necessari i vam haver de plantejar una planificació molt ajustada per realitzar el desenvolupament només al segon semestre. Tot i que es va intentar amb totes les forces, l'evidència i la realitat varen poder amb mi i com veurem en les següents pàgines, una serie de problemes inesperats en cada fase, varen fer impossible finalitzar com estava previst. Vam haver de replantejar l'esquema de planificació per acabar per setembre.

Algunes fases es van allargar degut a la dificultat de trobar els errors en el desenvolupament treballant amb dades reals tan grans. Moltes vegades la problemàtica estava en haver d'esperar 1 dia o 2 per obtenir uns resultats d'una modificació del codi, veure que estaven malament, modificar el codi, tornar a provar,...era bastant tediós. Per exemple, la planificació de la fase de automatització del càlcul de MUMs en comparacions eucariota es va incrementar d'una setmana que havíem planificat, a més d'un mes. Les raons van ser una planificació massa ajustada, sabíem que havíem d'adaptar els càlculs de MUMs a cromosomes i ens portaria temps, però havíem de pensar-ho així per finalitzar la feina per juny i no va ser possible.

Vam haver d'afegir una etapa de proves amb el visualitzador web de comparacions de genomes, el Mummy. Vam suposar que seria un fet immediat, però es van haver de fer proves i adaptacions per poder carregar els fitxers SMUMs. No obstant, altres fases com la paral·lelització i la adaptació del càlcul SMUMs van tenir la durada prevista.

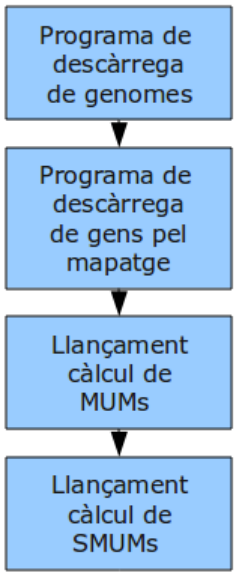



[Figura 11] Diagrama de Gantt de la planificació real.

## 3.2 Fase 1: Adquirir els coneixements bàsics

Al inici de qualsevol projecte on hi ha hagut treball realitzat abans de la teva incorporació, hi ha un fase inicial de posar-se al dia i aprendre tot el que s'ha fet, el com, el perquè, etc. Al meu cas, vaig incorporar-me a un projecte del IBB [9], dins de la línia d'investigació de comparació de genomes, que té prop de 10 anys d'antiguitat. És per això que considero aquesta fase important, i volia fer-ne menció, doncs van caldre moltes reunions amb el co-director Mario Huerta per situar-me, entendre la problemàtica a resoldre i proposar un seguit d'estratègies per abordar el problema. Aquestes estratègies coincideixen amb les fases que estic explicant i explicaré als següents apartats.

Després de la lectura de les publicacions relacionades [5][6][7][8], la lectura de tot el treball previ relacionat i les reunions de dubtes, vaig poder començar a veure codi i començar a fer proves amb la comparació de bacteris i estudiar com funcionava el codi. De primeres, va ser molt complicat i tot el procediment se'm feia una muntanya, però amb temps, paciència i part per part vaig poder anar desglossant tota la feina i entendre el funcionament de la comparació de genomes per bacteris. A continuació faré un petit resum del procediment de comparació de genomes per bacteris (també vàlid per arqueobacteris) [Figura 12].




[Figura 12] Procediment de comparació de genomes del arqueobacteris i bacteris.

-Petit programa que descarrega un fitxer de genomes complets comprimits del FTP del NCBI i els descomprimeix.

-Un altre programa que serà el mateix per eucariotes descarrega els gens pel seu mapatge sobre el genoma.

-Un programa s'encarrega de fer el llançament del càlcul de MUMs tots contra tots i de un en un.

-Finalment s'executa un programa que fa el llançament del càlcul dels SMUMs per cada fitxer de MUMs.

Un cop tenia el procediment clar per bacteris em vaig dedicar a estudiar i a repetir unes proves de comparació de genomes eucariotes, que es van realitzar al 2010 per demostrar que l'aplicatiu web per la visualització de comparacions, podia suportar la gran quantitat de dades que conté una comparació d'eucariotes. L'aplicatiu web va suportar perfectament la gran quantitat de dades, però com la comparació d'eucariotes no era part del treball no va quedar constància dels procediments i no es va mirar tant per la exactitud de la comparació realitzada.

Aquestes proves consistien en la comparació del Homo Sapiens contra el Macaca Mulatta. Em van servir per distingir les diferències principals.

Vaig recollir tot el treball realitzat i algunes idees per aplicar-les posteriorment al meu desenvolupament per la comparació de genomes eucariotes que continuaré explicant en detall a les següents seccions.

Abans de qualsevol automatització s'ha de saber exactament que és el que ha de fer el programa, així que després d'aquesta prova vaig decidir començar de cero, descarregar dos nous genomes eucariotes del FTP NCBI i realitzar el càlcul de MUMs manualment. Tot i les dificultats trobades, que esmentaré als apartats pertinents, els resultats van ser positius, però vaig veure el que d'entrada ja sabíem, la generació de MUMs trigava  més d'un dia. No podia perdre temps, i havia de automatitzar el procés el més aviat possible per calcular els MUMs a temps.

Aquí la decisió va ser llançar el càlcul de MUMs en quant tingués un procediment automatitzat vàlid, perquè anés calculant MUMs mentrestant jo treballava en la resta d'objectius. Així és com es van formar dos grans fases paral·leles, que envolten tot el treball:

-La fase de càlcul de MUMs de tots els mamífers disponibles.

-La fase de càlcul de MUMs de la resta d'animals disponibles.

Aquestes fases no les he contemplat a la planificació doncs eren càlculs que llançava al servidor i només els revisava de tant en tant. La importància d'aquestes fases, és que al càlcul dels MUMs per animals, vaig trobar multitud de problemes que em van obligar a adaptar diversos algorismes que ja donava per realitzats. Com es tractaven de petites adaptacions i per no complicar la planificació no han quedat representades, però volia fer menció per preparar al lector, ja que durant les següents seccions de vegades faré referència a problemes sorgits durant el càlcul de MUMs d'animals.

## 3.3 Fase 2: Automatització de la generació de MUMs

Era urgent posar-me a comparar mamífers i generar els MUMs d'aquestes comparacions. És per això que vaig començar a realitzar la generació de MUMs automatitzada. Suposem que ja tinc descarregat tot el necessari i puc fer la comparació, la part de la descarrega la veurem en més detall a la fase 6 en l'apartat d'automatització del robot de descarrega de nous genomes eucariotes. Per tant disposem de :

-Tots els cromosomes del genoma.
-Programa de càlcul de MUMs per bacteris

Sabíem el procediment a realitzar però abans d'automatitzar el procés, calia solucionar els problemes que hem esmentat al apartat anterior veiem-los un a un.

### 3.3.1 Concatenació de la seqüència dels cromosomes d'un genoma

El programa de càlcul de MUMs que disposem compara genoma contra genoma. Si volem comparar un genoma en cromosomes contra un altre genoma, necessitem concatenar les seqüències de cromosomes que hem descarregat. Perquè la seqüència completa del genoma tingués sentit, havíem de realitzar la concatenació de la seqüència dels cromosomes en ordre. Entre els cromosomes podem trobar amb diferents identificadors:

-Números: 1,2,3,4,...
-Especials: X,Y
-Especials numerats: X1,X2,Y5,...
-Números repartits en dos cromosomes: 2A,2B,3C

Vaig haver de desenvolupar un programa que obtingués els identificadors dels cromosomes per poder realitzar una ordenació correcta i generar genoma complet.

Em va preocupar veure que cada cromosoma disposava d'una línia identificativa del cromosoma, que començava per un coixinet(#) i podia alterar la comparació.

Observant el programa que genera els MUMs vaig detectar que ja estava preparat per ignorar aquest tipus de línies

Vaig haver d'aplicar una millora al programa de concatenació a la fase de comparació dels animals. El conjunt de identificadors possibles es va veure augmentat enormement al sortir dels mamífers. Ara a més dels anteriors tipus d'identificadors ens podíem trobar:

-Del tipus especials però amb més variacions: W,Z, LG1, LG2, LGE22,...
-Lletres minúscules: a,b,c,d,...
-Estranys: LG_B01,LG_B02, LG1=X, ...

Vaig haver de buscar una solució a aquesta anomalia del NCBI per escollir els noms dels identificadors de cromosomes. Vaig trobar que amb l'ús de expressions regulars [10] el meu problema era abordable i vaig poder contemplar tots els casos, i ordenar-los correctament. En cas que vingui un nou identificador d'estrany simplement s'afegiria al final per no perdre la informació.

### 3.3.2 Adaptació del càlcul de MUMs per cromosomes

Necessitàvem modificar el codi que llança el càlcul per la generació de MUMs per adaptar la comparació per cromosomes. Per bacteris es crida a una funció indicant els dos fitxers que han de ser comparats i crida al MUMOL per generar els MUMs. Havíem d'afegir en aquesta funció un bucle i per cada cromosoma del genoma , cridar al MUMOL indicant el cromosoma i el segon genoma. El cromosoma aniria canviant a cada volta i el segon genoma seria el mateix per tota la comparació.

A més havia de tenir en compte les zones no seqüenciades de qualsevol genoma o cromosoma. Són cadenes de N's a la seqüència que indiquen que encara no esta disponible aquesta part del genoma (pe. ACTNNNGTGNNAGTCAT). El algoritme MUMOL ignora aquestes parts per la comparació, però ha de guardar en un fitxer les posicions de inici i duració de la seqüència de N's tant pel cromosoma tant com pel genoma, per després al programa d'automatització corregeixi les posicions dels MUMs. Aquesta part estava contemplada per la comparació genoma contra genoma, però en la adaptació per cromosoma s'havia de realitzar prèvia a la concatenació perquè les posicions finals de cada genoma quedessin correctes.

### 3.3.3 Concatenació de MUMs de la comparació per cromosomes

Després de la generació de MUMs per cada cromosoma, i d'haver sumat els desplaçaments corresponents a les zones no seqüenciades, tenim un conjunt de fitxers de MUMs, un per cada cromosoma, que hem de concatenar si volem obtenir la comparació entre els dos genomes.

Per realitzar correctament aquesta concatenació de MUMs de cromosomes hem de tenir en compte que la generació de MUMs s'ha fet relativa al cromosoma, i no al genoma complet. Així doncs els fitxer de MUMs corresponents al cromosoma 1 i al cromosoma Y contra un genoma podria ser de la següent manera:

**genoma1-chr1_genoma2:**

| Posició inicial genoma 1 | Posició inicial genoma 2 | Longitud del MUM |
|---|---|---|
| 3 | 54 | 26 |
| 30 | 2 | 20 |
| ... | ... | ... |

**genoma1-chrY_genoma2:**

| Posició inicial genoma 1 | Posició inicial genoma 2 | Longitud del MUM |
|---|---|---|
| 3 | 28 | 26 |
| 30 | 158 | 20 |
| ... | ... | ... |

És fàcil adonar-se que els MUMs del cromosoma 1 i del cromosoma Y ocupen les mateixes posicions del genoma 1, però realment no es cert, hem comparat seqüències diferents. Així que el cromosoma 1 en aquest cas per ser el primer cromosoma tindria les posicions inicials dels MUMs correctes (relatives al genoma) però pels MUMs del cromosoma Y hauríem d'aplicar una desplaçament. Aquesta correcció consistiria en sumar sistemàticament a totes les posicions inicials del genoma 1 un desplaçament corresponent amb la posició que ocupa el cromosoma Y al genoma.

Per això vaig desenvolupar l'algorisme que desplaçava aquestes posicions mentre llegia els MUMs d'un cromosoma. El problema és evident, necessitava el valor de desplaçament de cada cromosoma.

El servidor del NCBI ofereix les posicions d'inici dels cromosomes però fer una consulta online era inviable, amb el que es necessitava una altra solució. El MUMOL parseja el cromosoma caràcter a caràcter i vaig trobar dos variables que emmagatzemaven el valor numèric dels caràcters llegits, un que no comptabilitzava les regions no seqüenciades (per obtenir la llargària de la seqüència útil) i un altra variable que comptabilitzava tots els caràcters Perfecte, ja tenim el desplaçament de cromosoma que necessitava. Després de comprovar que el desplaçament realment coincidia amb el desplaçament que apareixia a la web del NCBI, ja ho tenia segur.

Així doncs vaig afegir la funcionalitat al MUMOL per escriure un fitxer amb el identificador del cromosoma i el desplaçament que ocupa al genoma.

Només calia llegir el fitxer de desplaçaments, ordenar els cromosomes i concatenar els fitxers de MUMs de cromosomes en ordre i conforme generem el fitxer de MUMs de la comparació, corregim les posicions inicials sumant el desplaçament de tots els genomes anteriors. Suposant un desplaçament total de 100, el fitxer de MUMs resultant de l'exemple seria una cosa com aquesta:

**genoma1-_genoma2 :**

| Posició inicial genoma 1 | Posició inicial genoma 2 | Longitud del MUM |
|---|---|---|
| 3 | 54 | 26 |
| 30 | 2 | 20 |
| ... | ... | ... |
| 103 | 28 | 26 |
| 130 | 158 | 20 |

Igual que al programa de concatenació de la seqüència dels cromosomes d'un genoma, en aquest programa de concatenació de MUMs de cromosomes també vaig haver d'aplicar una adaptació a la fase de comparació dels genomes d'animals tots amb tots. Tots els programes que treballessin ordenant els identificadors dels cromosomes es van veure afectats en aquesta fase. El nou conjunt de identificadors possibles em va obligar a fer us de expressions regulars per concatenar adequadament els MUMs.

### 3.3.4 Eliminació de no-MUMs

Hi havia un altre petit problema de concepte en la comparació per cromosomes, la creació de no-MUMs als fitxers de MUMs. Situació: cada cop que un cromosoma es compara contra l'altre genoma sencer, està o pot estar generant MUMs per tot el cromosoma i per tot l'altre genoma; comparem el següent cromosoma i tornem a generar MUMs per tot el nou cromosoma i tornem a generar MUMs per tot l'altre genoma. No cal seguir, pot donar-se la casuística que el genoma complet tingui dos MUMs a la mateixa posició i al no ser seqüències úniques, ja no són MUMs perquè el nom ho indica, els MUMs han de ser coincidències màximes i úniques.

No hi ha cap alternativa, ja que la comparació ha de ser per cromosomes. Així que s'ha de fer un post-procés per eliminar aquests no-MUMs.

Primerament, vaig desenvolupar un algorisme que tot i solucionar el problema el vam desestimar per no ser eficient. L'algorisme feia el següent:

1-Ordenava el fitxer de MUMs per la posició del genoma 2, és a dir, a la columna on podíem trobar MUMs solapats i ordenava secundàriament per la tercera columna, per trobar els MUMs més grans abans al fitxer.
2-Després recorria cada posició d'inici de MUM del genoma 2 i mirava la longitud del MUM.
3-Aquest era el MUM actual i recorria tots el MUMs següents que el seu inici de posició estigues dintre del MUM actual.
4-Si el MUM començava i acabava dins del MUM actual es marcava per esborrar.
5-Un cop hem recorregut tots els MUMs de dintre del MUM actual, rebobinem el fitxer fins al MUM actual i marquem com a MUM actual el següent.
6-Repetim el procés fins acabar el fitxer.

Podem veure que era totalment ineficient per la quantitat de lectures que fem a disc, avançant i rebobinant el fitxer contínuament.

A una reunió amb en Mario Huerta, vam pensar de nou l'algorisme i vam donar amb un sistema moltíssim més eficient. Aquest és l'algorisme de eliminació de no-MUMs que s'ha aplicat al final.

1-Ordena el fitxer de MUMs per la posició 2, és a dir, a la columna on podem trobar MUMs solapats i ordena secundàriament per la tercera columna, per trobar els MUMs més grans abans al fitxer.
2-Després recorre cada posició d'inici de MUM del genoma 2 i la guarda en un vector dinàmic juntament amb la longitud del MUM i la posició del genoma 1. Aquesta llista són els MUMs possiblement absorbents Cada cop que entra un MUM al vector ja s'escriu al fitxer de sortida.
3-Per cada nou MUM que és processa, és comprova que no quedi solapat per qualsevol dels MUMs possiblement absorbents de la llista. Si és solapa, saltem al següent MUM, si no és solapa, l'afegim com a possible absorbent
4-Quan s'arriba al final del fitxer s'escriuen els MUMs que quedin a la llista i finalitza.

Podem observar que aquest algorisme és moltíssim més eficient i ràpid (unes 6-7 vegades més ràpid), ja que només hem de llegir el fitxer de MUMs un cop.

### 3.3.5 Optimització d'ús de memòria

Tenia ja tot el programa de calcul de MUMs per genomes eucariota automatitzat i 100% funcional. Vaig fer el llançament de generació de MUMs per tots els mamífers per tenir els resultats el més aviat possible.

Tot funcionava correctament fins que un dia l'execució va omplir la memòria RAM i va començar a fer servir memòria Swap. Això provoca que tot funcioni molt lent, tant el programa com la resta de servidor i vaig haver de finalitzar el programa per la força. No vaig saber que va passar, i em vaig veure obligat a afegir al programa un sistema de notificació de les comparacions de genomes que havien acabat correctament. També vaig afegir un fitxer de log. Vaig tornar a llançar la generació de MUMs i al temps va passar exactament el mateix, però aquest cop vaig poder esbrinar que passava ja que va fallar exactament al mateix lloc, però aquest cop tenia més dades.

Per fer-nos una idea del que passava, haig d'explicar les necessitats de memòria del algorisme MUMOL durant la comparació. El MUMOL ha de carregar en memòria un dels genomes i el representa amb una serie d'estructures en arbre. Aquestes estructures són el gruix de dades que més ocupen a memòria.

Tornem al nostre problema. Quan el programa es descontrolava i començava a fer servir Swap era perquè estava en curs la comparació del ornitorinc. Vaig veure que aquest genoma té un cromosoma de gairebé 500mb i quan es carregava l'estructura en memòria d'aquest cromosoma omplíem pràcticament tota la RAM que disposàvem. Aproximadament unes 120 vegades la mida del cromosoma, és a dir, uns 60Gb.

La solució al problema no va ser gaire complicada. Com només un dels genomes ha de carregar-se amb aquestes estructures tan grans, havia de comprovar el mida dels dos genomes i enviar la comparació de manera que el genoma més petit fos el que es comparés per cromosomes i generant les estructures pesades del MUMOL. D'aquesta manera totes les comparacions cabien i no només això sinó que vaig notar un descens de la memòria RAM utilitzada en totes les comparacions. A l'hora de la paral·lelització, podríem comparar més genomes alhora gracies a aquesta millora. Veurem a la següent apartat 3.4 de paral·lelització un sistema desenvolupat per detectar comparacions que no caben a memòria. En aquest cas la comparació dels genomes no es realitza i es crea un avís al administrador.

## 3.4 Fase 3: Paral·lelització del càlcul de MUMs

En aquesta fase, ja tenia un parell de execucions de càlcul de MUMs alhora en dos carpetes diferents i calculant genomes diferents per anar aprofitant el servidor. Havia d'afegir la paral·lelització aviat per dos raons, primer, per tenir un control de memòria i no excedir-me del ús que em pertocava i segon, volíem tenir a temps el calcul de MUMs de totes les eucariotes, així que fins que no tingués la paral·lelització no podia generar MUMs al ritme que em permetia el potent servidor [4].

Al cas de bacteris, la limitació a la paral·lelització és el numero de CPU's disponibles. Al cas de eucariotes hem de controlar no excedir-nos en l'ús de memòria RAM. Havia diversos problemes d'entrada:

-Saber de forma fiable la memòria disponible al servidor. On consultar-la i com?
-La necessitat de saber a priori la quantitat de memòria RAM que ocuparia la comparació de cada cromosoma contra l'altre genoma, evidentment, per saber si podem llançar la comparació o no.
-La lentitud del MUMOL per agafar la memòria que necessita. Potser una comparació d'un cromosoma-genoma necessita 20 Gb, doncs això és el que succeeix: comença l'execució reservant 1gb i fins que no porta 10 minuts d'execució no esta utilitzant els 20gb. És per això que haig de buscar alguna forma de fer el llançament paral·lel sense acabar-me la memòria.
-Solapament de fitxers amb el mateix nom i escritures desordenades en fitxers compartits.

### 3.4.1 Consulta de memòria RAM disponible

Hi han diverses maneres de consultar la informació de memòria en un sistema GNU/Linux. Existeixen comandes com "memfree" o "top" on podem veure l'ús de memòria del sistema en un moment determinat. La primera manera que se'm va acudir va ser escriure el resultat de la comanda en un fitxer i llegir els valors que m'interessaven, cosa que no era gens eficient ja que tractàvem d'evitar escriptures i lectures a disc sempre que fos possible. A més, en aquest cas es tractava d'una consulta que hauríem de fer sovint per anar llançant els cromosomes.

També disposava del fitxer /proc/meminfo que una simple lectura, sense haver de fer procediments rebuscats, em permetria obtenir el valor de memòria. Inicialment tampoc em va agradar aquest sistema ja que volia evitar l'ús de fitxers a disc, i havia d'existir alguna manera de fer una consulta d'aquestes característiques sense massa complicació.

Però no la vaig trobar i la raó va ser que el fitxer /proc/meminfo ja era la solució perfecta. Els sistemes GNU/Linux tenen al /proc tota una estructura de fitxers d'informació al sistema com processos del sistema, informació de CPU, de memòria com el fitxer meminfo, entre molts altres. Tota aquesta estructura de fitxers està carregada en memòria pel que els accessos a aquests fitxers són immediats i no cal anar a disc dur.

Vaig fer una funció per llegir el fitxer /proc/meminfo i obtenir el valor de memòria lliure. Veiem el format del fitxer [Figura 13]:

```
$ cat /proc/meminfo
MemTotal:     65997056 kB
MemFree:       8298664 kB
Buffers:         49348 kB
Cached:       15539884 kB
SwapCached:      15608 kB
Active:       55051496 kB
Inactive:      2023596 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:     65997056 kB
LowFree:       8298664 kB
SwapTotal:    204901032 kB
SwapFree:     204818776 kB
Dirty:            20 kB
Writeback:           0 kB
AnonPages:    41485212 kB
Mapped:          15720 kB
Slab:           389012 kB
PageTables:      94292 kB
NFS_Unstable:        0 kB
Bounce:              0 kB
CommitLimit:  237899560 kB
Committed_AS: 45309436 kB
VmallocTotal: 34359738367 kB
VmallocUsed:    363576 kB
VmallocChunk: 34359374779 kB
HugePages_Total:     0
HugePages_Free:      0
HugePages_Rsvd:      0
Hugepagesize:     2048 kB
```

[Figura 13] Contingut del fitxer /proc/meminfo

La funció havia de llegir el fitxer línia per línia i identificar el camp i el valor. D'aquesta manera ja teníem el valor de memòria lliure, però la cosa no acabava aquí.

Em vaig adonar que el valor de memòria lliure sempre era molt baix, semblava que el sistema sempre tenia la memòria RAM ocupada i per cada comparació que fèiem, la memòria quedava plena i no es buidava. Vaig haver de buscar mètodes per buidar la memòria "cached". Vaig trobar la comanda [11]:

```
# sync ; echo 3 > /proc/sys/vm/drop_caches
```

El que feia aquesta comanda era precisament alliberar aquesta memòria i el valor de memòria lliure es reflectia correctament. Vaig programar una tasca de GNU/Linux amb el cron per executar cada dia aquesta comanda.

En aquell moment no ho sabia però estava perdent el tems, i em va servir per aprendre un parell de coses. Casualment, als pocs dies vaig trobar una pàgina web [12] d'informació del sistema operatiu android que explicava les causes de perquè no era bo buidar la memòria cache, deia així:

"Unused RAM is wasted RAM (memoria no utilizada és memoria desperdiciada) es la filosofía que apropia este Sistema Operativo móvil, que también es usada en Windows 7 y Gnu/Linux. Por lo anterior Android es capaz de utilizar casi el 100% del total de la memoria RAM." [12]

Vaig comprovar que això era cert i vaig comprovar que el sistema alliberava la memòria que no utilitzava conforme la necessitava així que aquest no era un problema per realitzar la comparació i que no havia de fer la "neteja diaria" de memòria RAM.

De nou estava com al principi, no tenia el valor de memòria lliure, ja que era un valor molt petit per culpa de la cached memory. Havia de fer el càlcul, buscar la fórmula per obtenir el valor de memòria lliure real i disposava de tots els elements al fitxer meminfo [Figura 13]. A més, havia de contemplar que el servidor no era d'ús exclusiu de la meva aplicació, així que es va acordar definir un limit màxim de RAM que podia fer servir, actualment 52Gb aproximadament. El càlcul resultant de la memòria va ser el següent:

Memòria ocupada real = (MemTotal-MemFree-Cached-Buffers)
Memòria ocupada i no disponible = Memòria ocupada real + (MemTotal-52Gb)
**Memòria lliure = MemTotal – Memòria ocupada i no disponible**

Vaig fer les proves pertinents i vaig veure que tant fent el càlcul anterior i alliberant la RAM que no es feia servir, la memòria disponible real coincidia.

### 3.4.2 Preveure l'ocupació de memòria d'una comparació

Ara ja sabíem la memòria RAM disponible i calia saber si podíem llançar una comparació d'un cromosoma més en paral·lel o calia esperar.

El MUMOL genera unes estructures per comparar els genomes que ocupen aproximadament unes 120 vegades la mida del primer genoma, al nostre cas el cromosoma que comparem. A aquest valor d'ocupació de memòria, l'hauríem de sumar la mida del segon genoma que es carrega sencer a memòria. Exemple:

Comparació de genoma 1-cromosoma 5 (100Mb) i genoma 2 (3,5Gb)
100Mb*120+3584Mb = 15584Mb = 15,22Gb

Llavors podia saber a priori si una comparació es podrà realitzar o faltarà memòria, així que vaig afegir un control abans de fer cap càlcul, que mirava si el primer cromosoma, el més gran, podia cabre a memòria, sinó, saltava la comparació i avisàvem al administrador.

### 3.4.3 Llançament de cromosomes en paral·lel

Havia de trobar el millor mètode per llançar el màxim número de comparacions de cromosomes alhora. Hem vist al apartat anterior com calcular el valor que ocuparà una comparació, necessitava construir una llista dinàmica i afegir el nom del cromosoma juntament amb el valor de memòria que necessita per ésser comparat. Aquesta llista es creava en ordre decreixent, tenint els cromosomes més grans al principi. A l'algorisme a desenvolupar havíem de tenir cura de la reserva progressiva de memòria que feia el MUMOL al executar-se. Vaig optar per fer la mesura de la memòria lliure al inici del bucle, i després fer ús d'una variable de memòria que s'anava decrementant i restant cada cop la mida de la comparació de cromosoma-genoma llançada.

L'algorisme desenvolupat feia el següent:

-Bucle que s'executa mentre quedin cromosomes a la llista.
-Comprovem la memòria disponible i la guardem en una variable.
-Seguidament recorrem tota la llista de cromosomes en ordre, primer els més grans.
-Si el cromosoma actual cap en memòria, l'eliminem de la llista i creem un nou procés (crida fork) que llança els càlculs de MUMs pel cromosoma.
-El procés principal decrementa l'espai que ocupa la comparació llançada a la variable de memòria disponible.
-Continuem recorrent la llista llançant en paral·lel totes les comparacions fins que o bé s'acabi la llista o bé no quedi més "s'esgoti" la memòria en la variable de memòria disponible.
-Fem una crida wait [14] per esperar els fills i quan el procés principal pugui continuar, voldrà dir que ja hi ha memòria disponible. Tornem al inici del algorisme.
-Fi de l'algorisme.

Vaig decidir afegir un control màxim de llançament de comparacions de cromosomes alhora. Aquest servidor [4] ha de realitzar tant les comparacions de genomes eucariotes com les comparacions de bacteris i arqueobacteris. Aquestes ultimes necessiten molt poca memòria RAM però molts processadors. Per no haver de competir pels processadors vaig incloure una variable de màxims processos que evita que es llencin més de 10 comparacions (Si el genoma es petit amb molts cromosomes es pot donar el cas que es llencin més.).

### 3.4.4 Adaptació del càlcul de MUMs a la paral·lelització

A l'últim pas vaig necessitar modificar el MUMOL per afegir l'identificador del cromosoma a cada fitxer que generava per la comparació. Si no realitzava aquesta adaptació, els fitxers es solapaven perdent així la informació que contenien. Els fitxers que necessitaven ser reanomenats eren la sortida de MUMs i els dos fitxer de desplaçaments de posicions del genoma no seqüenciades (seqüències de N's).

Els fitxers que eren relatius a tota la comparació sencera (és a dir no només a la comparació del genoma) teníem un problema semblant, i és que els diversos processos escrivien alhora als fitxers, obtenint dades inútils. Una opció era la de renombrar també aquests fitxers i al final de la comparació concatenar-los. Però es podia evitar i fer-ho d'una manera més elegant, utilitzant semàfors [13].

Havia de definir com a recursos els fitxers que ens interessava protegir, després sol·licitar i inicialitzar el semàfor al sistema operatiu. Els semàfors funcionen de la següent manera:

-El semàfor d'un recurs té una variable entera, on el valor simula el numero de operacions que ha de permetre alhora fins bloquejar-se.

-Quan sol·licitem escriure al fitxer, el semàfor es decrementa. Com l'inicialitzem a 1, només permet una escriptura alhora.

-Tots els processos que demanin escriure ara, quedaran bloquejats.

-Quan el primer acabi d'escriure, el semàfor s'incrementa en 1 i busca a la llista de processos bloquejats per continuar en ordre amb la següent escriptura.

-Així successivament.

Els fitxers de la comparació de genomes sencera són el fitxer de desplaçaments de cromosomes (per poder concatenar els MUMs de cromosomes i generar el MUM de la comparació) i els fitxer que guarden la similitud dels dos genomes.

## 3.5 Fase 4: Adaptació del procés de generació de SMUMs a eucariotes

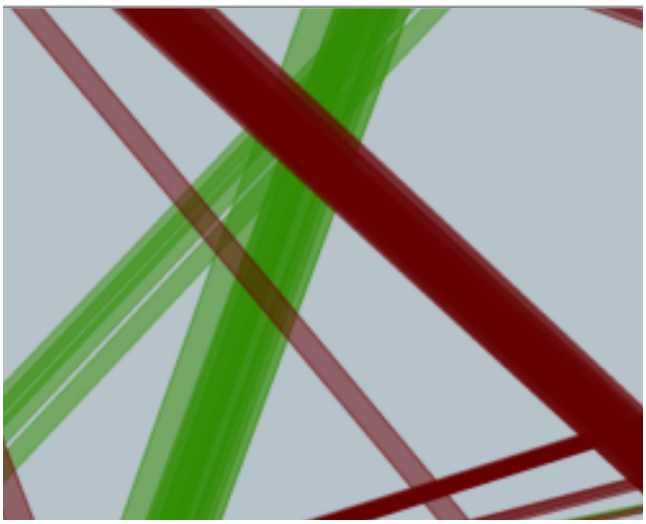
En aquesta fase, ja tenim els MUMs calculats i havia de realitzar la generació de SMUMs. La manera de fer-ho no estava clara. Teníem diferents possibilitats teòriques per generar fitxers de SMUMs, una era variant el multiplicador de gap escollit, i l'altre era variant el número de passades del algorisme.

Em van caldre multitud de proves amb diferents multiplicadors de gap, diferents números de passades de l'algorisme i lectura de tota la documentació sobre el tema per començar a entendre el funcionament del sistema de generació de SMUMs.

Primerament, tractava de fer tres passades de l'algorisme de SMUMs però l'arxiu que sortia, el tornava a passar al algorisme sense variar el format. Això provocava penjades del programa i que trigués molt més temps del normal. Llavors, va ser quan vaig veure les necessitats del programa de generació de SMUMs, necessitava un fitxer de MUMs, ordenat per la primera columna. D'entrada el fitxer que surt no pot ésser considerat un fitxer de MUMs, per dos raons, la primera és que les seqüències poden no ser úniques, i segona el fitxer no es troba ordenat per la primera columna. El fitxer de SMUMs conté tots el SMUMs ordenats per la primera columna, una línia de separació en blanc i continua amb els MUMs no absorbits per cap SMUM.

Per automatitzar el procés i solucionar aquest problema vaig fer un script que cridava al programa de generació de SMUMs i al finalitzar, eliminava l'espai en blanc del fitxer resultant i ordenava per la primera columna, tal com ho necessita l'algorisme de SMUMs. D'aquesta manera estem barrejant SMUMs i mums no absorbit, amb el que podem generar un fitxer amb "SMUMS i MUMs" que encara poden no ser únics. De moment em servia i vaig continuar les proves amb diferents multiplicadors i passades.

Els resultats d'aquesta etapa eren millors, el programa funcionava perfectament, però l'algorisme generava multitud de SMUMs repetits i que no ens servien [Figura 14].

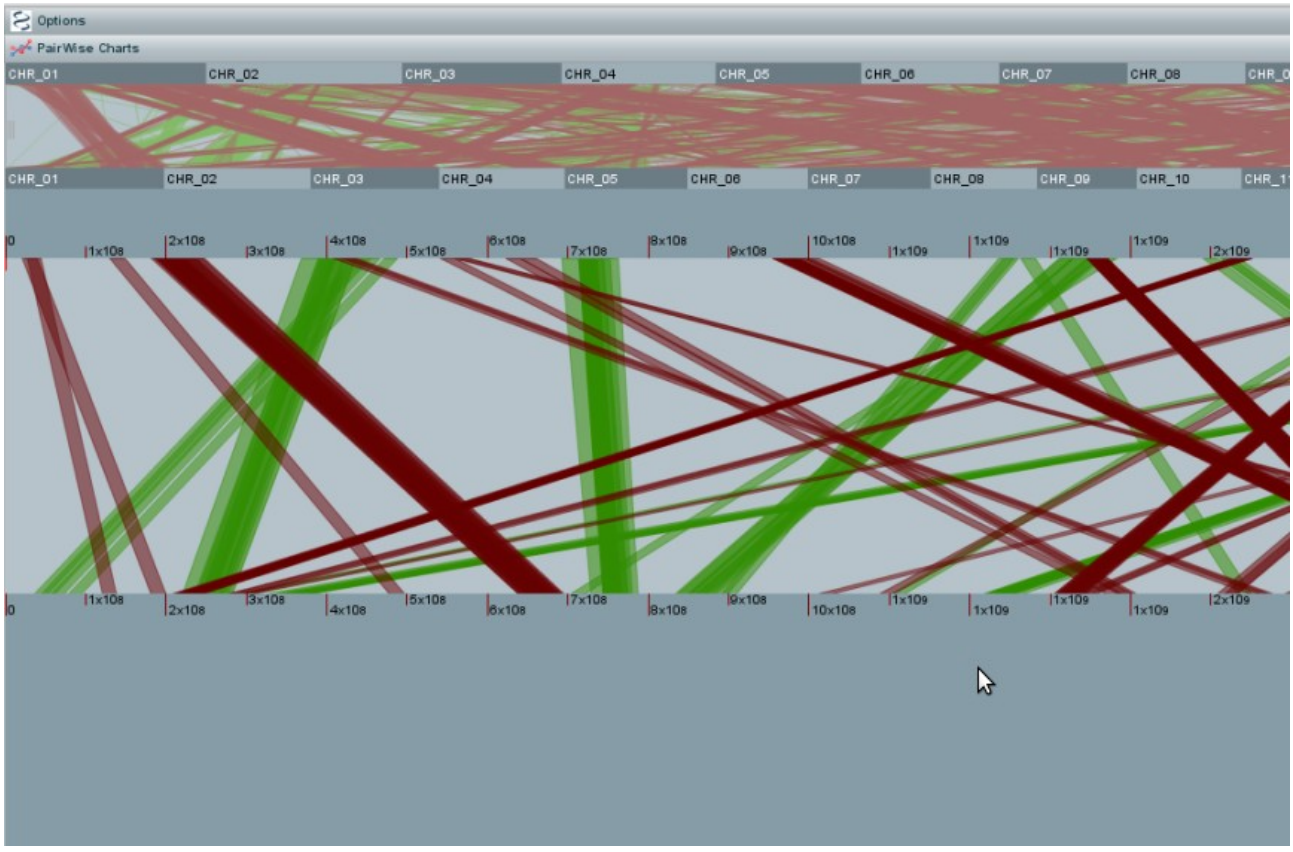


[Figura 14] Detall de SMUMs a l'applet web Mummy, podem observar els SMUMs directes (color verd) duplicats varies vegades i solapats.

Aquí vam decidir, a més de eliminar l'espai dels fitxers de SMUMs i ordenar per la primera columna, aplicar un algorisme que s'encarregués d'eliminar aquests SMUMs duplicats que es generen. Si a cada passada del càlcul de SMUMs eliminava els duplicats, el càlcul resultava més exacte i no perdíem dades.

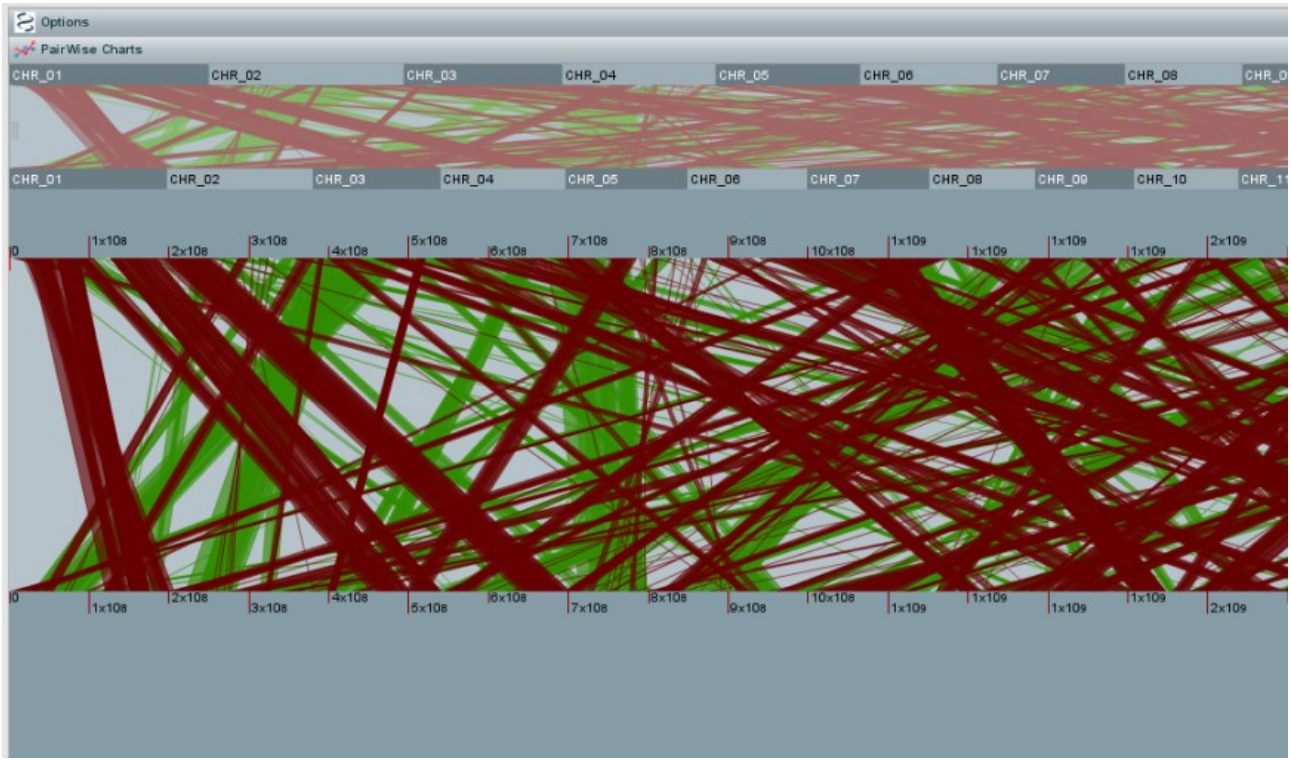
L'algorisme utilitzat, va ser una modificació del algorisme de eliminació de no-MUMs, però que es passava dos vegades, una per cada genoma, eliminant els SMUMs duplicats de les dues bandes.

La generació de SMUMs va començar a funcionar com la seda, però ja no vaig poder tornar a carregar fitxers a l'aplicatiu Mummy. Després de molt temps de proves i modificacions, el problema va ser que al aplicar la correcció de duplicats de SMUMs al final de tot, ens carregàvem el format SMUM del fitxer pel que applet Mummy no podia reconèixer el format. Si no realitzava la correcció de duplicats final els resultats eren desastrosos i es perdia molta informació [Figura 15].



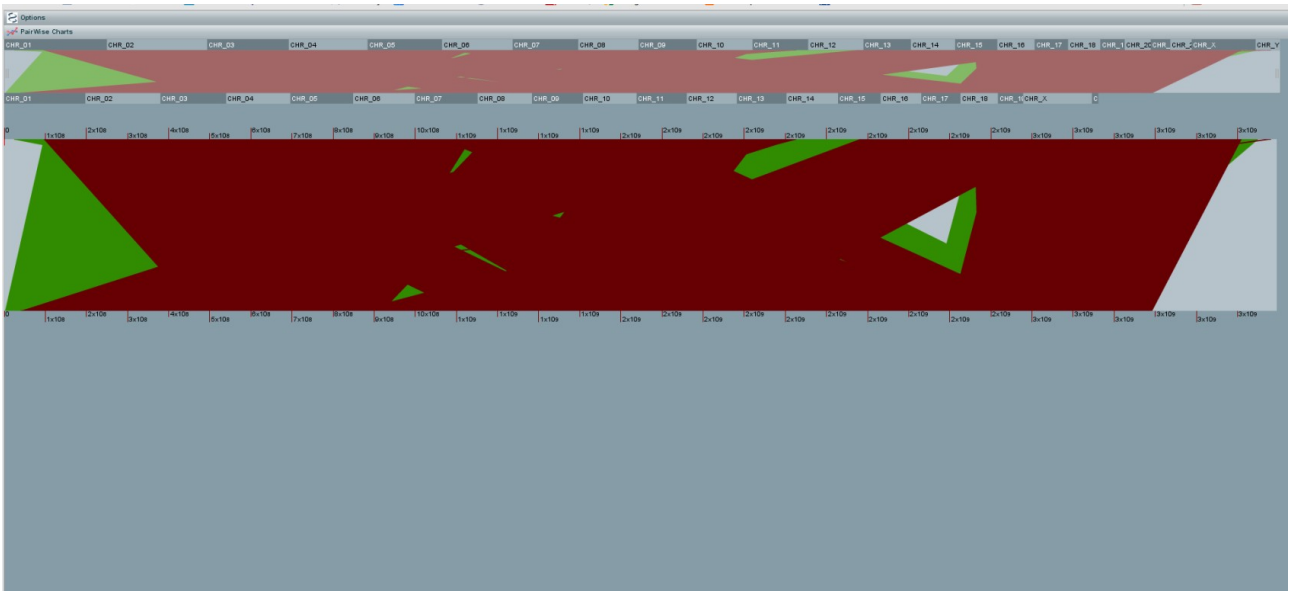

[Figura 15] Detall de SMUMs a l'applet web Mummy, podem observar que per culpa de no eliminar els duplicats a l'ultima passada perdem molta informació.

Havia d'aconseguir eliminar els SMUMs duplicats de l'ultima passada de SMUMs sense perdre el format, així que vaig realitzar un script que separava els SMUMs i els MUMs no absorvits, corregia els SMUMs duplicats i tornava a generar el fitxer amb el format de SMUMs. Ara veiem la diferencia [Figura 16].



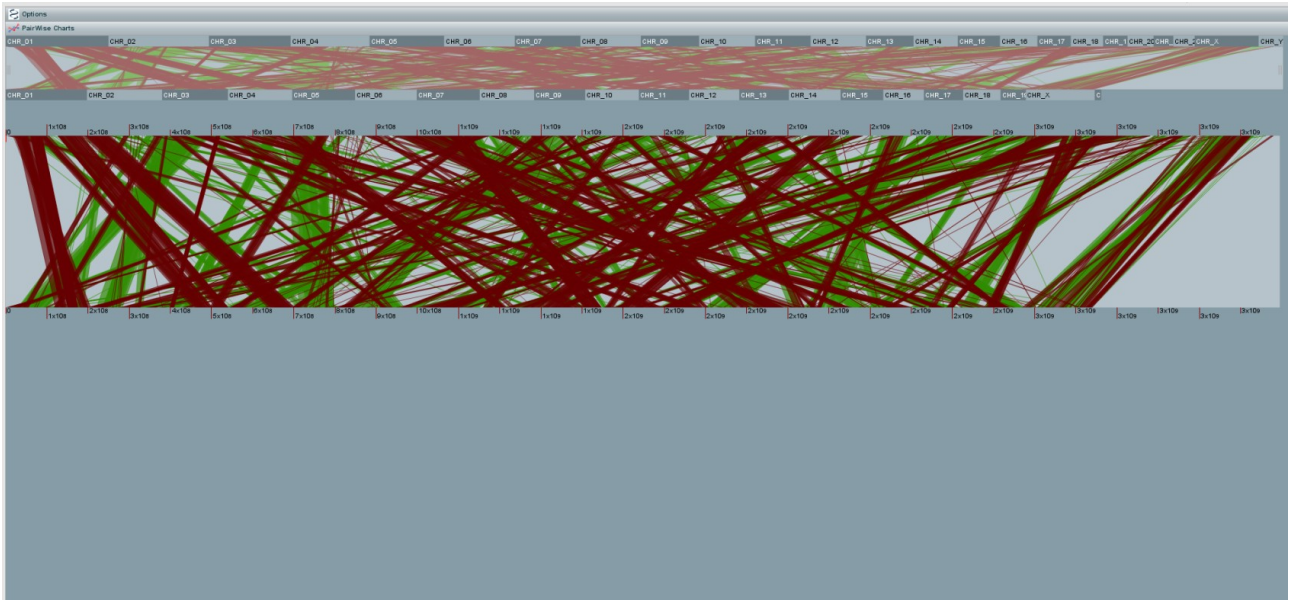

[Figura 16] Detall de SMUMs a l'applet web Mummy, podem observar que amb la millora d'eliminació de duplicats de SMUMs, a l'ultima passada millorem notablement els resultats.

Ara que el procediment funcionava correctament les proves de diferents passades i variacions de multiplicador tenien més sentit i es notaven les diferencies. Vaig fer multitud de combinacions, però a continuació veurem les més representatives. La primera que veurem és la primera passada amb multiplicador 300, segona passada amb multiplicador 50 i tercera passada amb multiplicador 25 (300-50-25), veiem el resultat a la [Figura 17]. Fixem-nos que el resultat no ens interessa doncs els SMUMs s'han agrupat en poquíssim SMUMs i hem perdut molta informació.



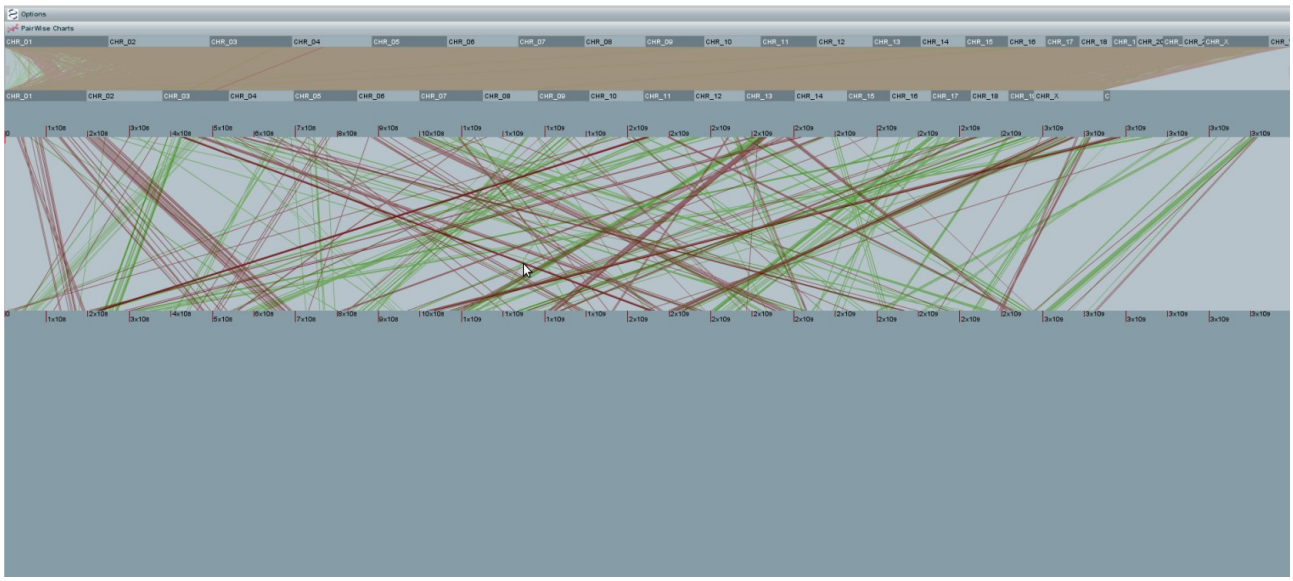
[Figura 17] SMUMs a l'applet web Mummy, generació de SMUMs amb 300-50-25, veiem que el resultat és molt dolent doncs els SMUMs s'han agrupat tots i hem perdut molta informació.

La següent imatge que veurem és la resultant de tres passades amb multiplicador 30 (30-30-30), veiem el resultat [Figura 18]. Aquest serà el resultat més correcte que vaig veure. Els SMUMs van creixent unint tots els MUMs més propers, sense ajuntar-se al final tots amb tots.



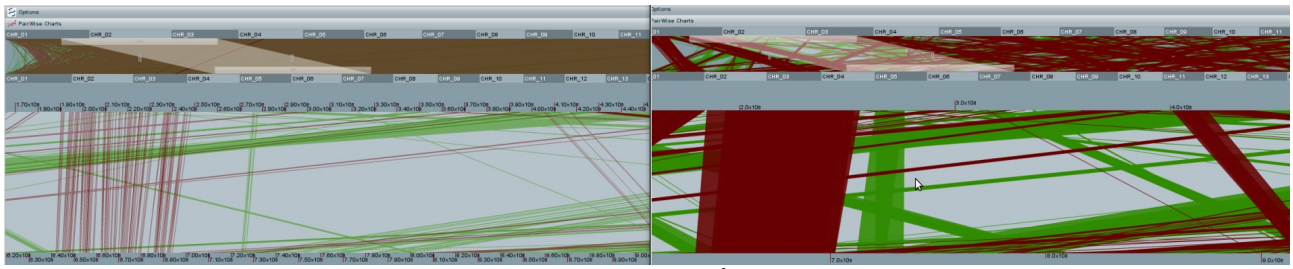
[Figura 18] SMUMs a l'applet web Mummy, generació de SMUMs amb 30-30-30, veiem un bon resultat, els SMUMs s'han unit sense provocar SMUMs exageradament grans.

La següent imatge és la corresponent a una única passada, amb multiplicador 100 [Figura 19]. Observem com en aquest cas no aconseguim ajuntar els SMUMs com s'han d'unir, veiem moltes línies juntes que haurien de formar un SMUM i no ho fan.



[Figura 19] SMUMs a l'applet web Mummy, generació de SMUMs amb multiplicador 100. El resultat no és incorrecte, però no hem aconseguit unir els SMUMs.

Finalment, observem aquesta figura [Figura 20] en detall on podem visualitzar una comparació de dos imatges, on es veu com s'ajunten els SMUMs correctament. A l'esquerra amb SMUMs generats amb una passada de 100 i a la dreta amb tres passades de 30-30-30



[Figura 20] SMUMs a l'applet web Mummy, comparació del mateix fitxer amb diferents passades, esquerra amb una passada de 100, dreta amb 3 passades de 30. Observem com s'ajunten els MUMs propers.

Totes les figures que hem observat, són els fitxers de SMUMs carregats a l'applet Mummy del servidor. Però els fitxers carregats necessiten una adaptació perquè l'applet els pugui carregar. L'algorisme per fer aquesta adaptació el va fer el desenvolupador del Mummy i consisteix en el següent:

-Primerament agafem el milió de SMUMs i MUMs més grans, per no sobrecarregar l'aplicació web.

-A continuació necessita afegir uns index d'ordenació per l'inici del SMUM al segon genoma, i un altre index d'ordenació per la mida dels SMUMs. Aquests index ajuden al Mummy a accedir i gestionar ràpidament les dades, sense perdre temps durant l'execució.

## 3.6 Fase 5: Automatització del programa de mapatge de gens

L'automatització del programa de mapatge no semblava gaire complicada. Estava programada per ser una aplicació independent de la resta. A l'execució es descarregaven els gens dels genomes demanats i generaven els fitxers de gens per cadascun.

El que si havia de fer, era adjuntar la crida a aquest programa, a la descarrega dels genomes, si havien actualitzacions. S'havia de tenir cura de proporcionar al programa tot el que necessita, un fitxer de text amb el nom dels genomes a descarregar els gens i per cadascun d'ells generar una carpeta amb un fitxer dintre amb els identificadors dels cromosomes i les Ids de dos bases de dades diferents del NCBI (Accession_version i RefSeq gi). El programa utilitza aquests identificadors per descarregar els arxius de gens que necessita i fer la unió correcta. Veiem el fitxer del que estem parlant [Figura 21]:

| #Chr | Accession.version | RefSeq gi | GenBank | GenBank gi |
|---|---|---|---|---|
| 1 | NC_000001.10 | 224589800 | CM000663.1 | 224384768 |
| 2 | NC_000002.11 | 224589811 | CM000664.1 | 224384767 |
| 3 | NC_000003.11 | 224589815 | CM000665.1 | 224384766 |
| 4 | NC_000004.11 | 224589816 | CM000666.1 | 224384765 |
| 5 | NC_000005.9 | 224589817 | CM000667.1 | 224384764 |
| 6 | NC_000006.11 | 224589818 | CM000668.1 | 224384763 |
| 7 | NC_000007.13 | 224589819 | CM000669.1 | 224384762 |
| 8 | NC_000008.10 | 224589820 | CM000670.1 | 224384761 |
| 9 | NC_000009.11 | 224589821 | CM000671.1 | 224384760 |
| 10 | NC_000010.10 | 224589801 | CM000672.1 | 224384759 |
| 11 | NC_000011.9 | 224589802 | CM000673.1 | 224384758 |
| 12 | NC_000012.11 | 224589803 | CM000674.1 | 224384757 |
| 13 | NC_000013.10 | 224589804 | CM000675.1 | 224384756 |
| 14 | NC_000014.8 | 224589805 | CM000676.1 | 224384755 |
| 15 | NC_000015.9 | 224589806 | CM000677.1 | 224384754 |
| 16 | NC_000016.9 | 224589807 | CM000678.1 | 224384753 |
| 17 | NC_000017.10 | 224589808 | CM000679.1 | 224384752 |
| 18 | NC_000018.9 | 224589809 | CM000680.1 | 224384751 |
| 19 | NC_000019.9 | 224589810 | CM000681.1 | 224384750 |
| 20 | NC_000020.10 | 224589812 | CM000682.1 | 224384749 |
| 21 | NC_000021.8 | 224589813 | CM000683.1 | 224384748 |
| 22 | NC_000022.10 | 224589814 | CM000684.1 | 224384747 |
| X | NC_000023.10 | 224589822 | CM000685.1 | 224384746 |
| Y | NC_000024.9 | 224589823 | CM000686.1 | 224384745 |
| MT | NC_012920.1 | 251831106 | J01415.2 | 113200490 |

[Figura 21] Fitxer Accessions original, veiem els identificadors que ens interessen a les tres primeres columnes.

Aquest que acabem de veure és el fitxer accession que s'acostumava a utilitzar per bacteris. Em vaig trobar amb el problema que aquest fitxer en eucariotes només era disponible als genomes més importants.

Així que vaig buscar alternatives i vaig trobar el fitxer chr_NC_gi que sempre existia a tots els genomes. Les dues primeres columnes eren iguals, justament els identificadors que necessitàvem. Veiem aquest fitxer [Figura 22]:

| #Chr | Accession.ver | gi | Assembly and unit | Assembly-unit accession.version |
|---|---|---|---|---|
| 1 | NC_000001.10 | 224589800 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 2 | NC_000002.11 | 224589811 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 3 | NC_000003.11 | 224589815 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 4 | NC_000004.11 | 224589816 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 5 | NC_000005.9 | 224589817 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 6 | NC_000006.11 | 224589818 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 7 | NC_000007.13 | 224589819 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 8 | NC_000008.10 | 224589820 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 9 | NC_000009.11 | 224589821 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 10 | NC_000010.10 | 224589801 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 11 | NC_000011.9 | 224589802 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 12 | NC_000012.11 | 224589803 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 13 | NC_000013.10 | 224589804 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 14 | NC_000014.8 | 224589805 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 15 | NC_000015.9 | 224589806 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 16 | NC_000016.9 | 224589807 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 17 | NC_000017.10 | 224589808 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 18 | NC_000018.9 | 224589809 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 19 | NC_000019.9 | 224589810 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 20 | NC_000020.10 | 224589812 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 21 | NC_000021.8 | 224589813 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| 22 | NC_000022.10 | 224589814 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| X | NC_000023.10 | 224589822 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| Y | NC_000024.9 | 224589823 | GRCh37.p2 Primary Assembly | GCF_000001305.13 |
| MT | NC_012920.1 | 251831106 | GRCh37.p2 non-nuclear | GCF_000006015.1 |
| 7 | AC_000068.1 | 89161212 | CRA_TCAGchr7v2 Primary Assembly | GCF_000000255.2 |
| 1 | AC_000133.1 | 157704448 | HuRef Primary Assembly | GCF_000000025.1 |
| 2 | AC_000134.1 | 157724517 | HuRef Primary Assembly | GCF_000000025.1 |
| 3 | AC_000135.1 | 157731950 | HuRef Primary Assembly | GCF_000000025.1 |
| 4 | AC_000136.1 | 157734150 | HuRef Primary Assembly | GCF_000000025.1 |
| 5 | AC_000137.1 | 157734151 | HuRef Primary Assembly | GCF_000000025.1 |
| 6 | AC_000138.1 | 157734152 | HuRef Primary Assembly | GCF_000000025.1 |
| 7 | AC_000139.1 | 157734172 | HuRef Primary Assembly | GCF_000000025.1 |
| 8 | AC_000140.1 | 157734173 | HuRef Primary Assembly | GCF_000000025.1 |
| 9 | AC_000141.1 | 157734174 | HuRef Primary Assembly | GCF_000000025.1 |
| 10 | AC_000142.1 | 157704449 | HuRef Primary Assembly | GCF_000000025.1 |
| 11 | AC_000143.1 | 157704452 | HuRef Primary Assembly | GCF_000000025.1 |
| 12 | AC_000144.1 | 157704453 | HuRef Primary Assembly | GCF_000000025.1 |
| 13 | AC_000145.1 | 157704454 | HuRef Primary Assembly | GCF_000000025.1 |
| 14 | AC_000146.1 | 157704455 | HuRef Primary Assembly | GCF_000000025.1 |
| 15 | AC_000147.1 | 157709134 | HuRef Primary Assembly | GCF_000000025.1 |
| 16 | AC_000148.1 | 157713457 | HuRef Primary Assembly | GCF_000000025.1 |
| 17 | AC_000149.1 | 157713538 | HuRef Primary Assembly | GCF_000000025.1 |
| 18 | AC_000150.1 | 157715044 | HuRef Primary Assembly | GCF_000000025.1 |
| 19 | AC_000151.1 | 157718668 | HuRef Primary Assembly | GCF_000000025.1 |
| 20 | AC_000152.1 | 157726890 | HuRef Primary Assembly | GCF_000000025.1 |
| 21 | AC_000153.1 | 157728228 | HuRef Primary Assembly | GCF_000000025.1 |
| 22 | AC_000154.1 | 157729478 | HuRef Primary Assembly | GCF_000000025.1 |
| X | AC_000155.1 | 157734237 | HuRef Primary Assembly | GCF_000000025.1 |
| Y | AC_000156.1 | 157734238 | HuRef Primary Assembly | GCF_000000025.1 |
| 1 | AC_000044.1 | 89161184 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 2 | AC_000045.1 | 89161198 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 3 | AC_000046.1 | 89161204 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 4 | AC_000047.1 | 89161206 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 5 | AC_000048.1 | 89161208 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 6 | AC_000049.1 | 89161209 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 7 | AC_000050.1 | 89161211 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 8 | AC_000051.1 | 89161214 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 9 | AC_000052.1 | 89161215 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 10 | AC_000053.1 | 89161186 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 11 | AC_000054.1 | 89161188 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 12 | AC_000055.1 | 89161189 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 13 | AC_000056.1 | 89161191 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 14 | AC_000057.1 | 89161192 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 15 | AC_000058.1 | 89161193 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 16 | AC_000059.1 | 89161194 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 17 | AC_000060.1 | 89161195 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 18 | AC_000061.1 | 89161196 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 19 | AC_000062.1 | 89161197 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 20 | AC_000063.1 | 89161200 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 21 | AC_000064.1 | 89161201 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| 22 | AC_000065.1 | 89161202 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| X | AC_000066.1 | 89161217 | Hs_Celera Primary Assembly | GCF_000000015.2 |
| Y | AC_000067.1 | 89161219 | Hs_Celera Primary Assembly | GCF_000000015.2 |

[Figura 22] Fitxer chr_NC_gi, veiem els identificadors que ens interessen a les tres primeres columnes.

Aquest fitxer conté moltes més entrades, per cada versió del genoma estan els identificadors de cada cromosoma. Això feia que el programa no funcionés així que vaig haver de modificar el programa perquè ignores les línies restants que no necessitàvem

L'adaptació en la primera fase de comparació de mamífers contra mamífers va ser molt més ràpid del que havíem planejat, tot va funcionar perfectament a la primera i vam guanyar moltíssim temps. Però en la fase de comparació de tots els animals, van aparèixer moltíssims problemes, com l'aparició de nous identificadors de cromosomes i finalment vaig haver de dedicar-li molt més temps del que havíem planejat al inici. Entre els problemes a adaptar del programa de mapatge de gens esmentaré:

-Adaptació del identificador de cromosoma del fitxer accession al nom del fitxer descarregat de la base de dades NCBI (gbs).
-Ignorar fitxers de gens de cromosomes incomplets o en proves i que no apareixen al fitxer accessions.
-Afegir funcionalitats al parser del mapatge de gens

### 3.6.1 Adaptació del identificador de cromosoma accession al nom del fitxer de gens descarregat corresponent al mateix cromosoma

Al nostre programa, es descarreguen tots els fitxers de gens que hi ha a la carpeta del genoma al FTP del NCBI[2]. Amb l'informació que es troba dintre, i els identificadors que es troben al fitxer accession pot fer consultes per descarregar el mapatge dels gens. A la fase de comparació d'animals entre ells, quan vaig trobar tots aquests problemes, els cromosomes tenen noms que surten de la normalitat, llavors es produeixen situacions inesperades. Els gens descarregats del FTP del NCBI [2] tenien noms del tipus CHR_01 i al accession apareixien com CHR_1 al no coincidir els noms el programa no trobava el fitxer i tancava l'execució. Això era el cas bàsic d'error, però podia sorgir amb els cromosomes estranys del tipus LG_B01, i s'havia de contemplar qualsevol casuística

Quan el programa falla no sabia exactament perquè era, llavors la complicació estava en trobar l'error que acabo d'explicar. Després de fer un debug, situar el problema i explotar la facilitat de Java per tractar strings, vaig poder contemplar els casos, i en cas de no trobar els fitxers, afegir un cero a la posició correcta per trobar-ho, etc.

### 3.6.2 Ignorar fitxers de gens de cromosomes incomplets o en proves i que no apareixen al fitxer accessions

En el moment de descarrega dels fitxers de gens del FTP del NCBI [2] es descarreguen totes les carpetes de cromosomes que hi hagin. El problema és que hi han genomes que tenen cromosomes incomplets o en proves i que no apareixen al fitxer accessions perquè no els podem situar al mapatge de gens. Quan el programa intentava gestionar-los no trobava els fitxers corresponents a les seqüències de mapatge (per no aparèixer al accession) i el programa finalitzava la execució.

De nou havia de realitzar l'execució pas per pas fins trobar l'error i identificar-lo. Havia dos opcions, una era no descarregar els fitxers d'aquests cromosomes incomplets, però la opció més senzilla era fer una comprovació de si el fitxer que volem accedir existeix. Si no existeix vol dir que no apareix al fitxer accession i que no l'hem de tractar.

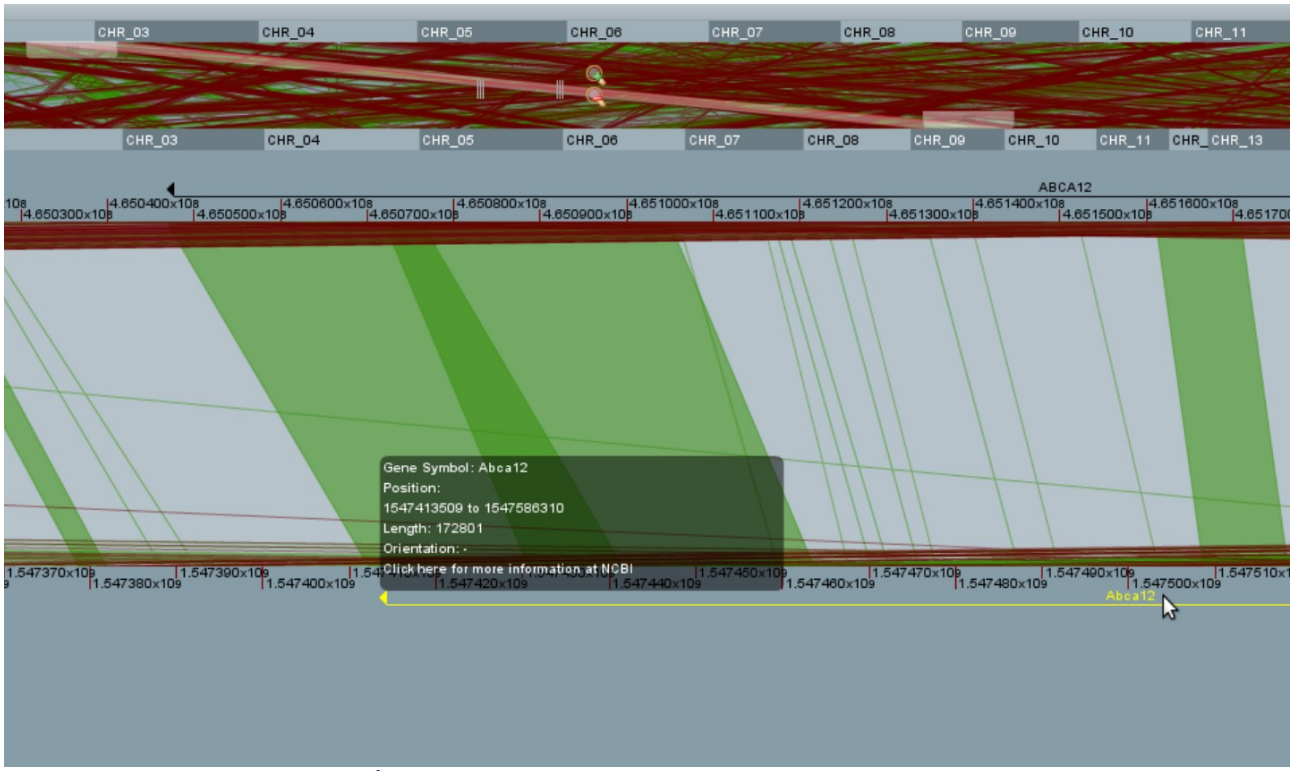
### 3.6.3 Millora del parser de mapatge de gens

Un últim cop més vaig haver de executar pas per pas el programa per trobar un error d'un tancament inesperat. Resultava que a la part final del programa, on es fa el parseig del fitxer de mapatge de gens, existeixen una serie de paraules clau que el programa detecta i guarda els espais a deixar entre gens, i el identificador del gen per fer la consulta online amb tota la informació del gen des de el Mummy. Als nous genomes d'animals, van aparèixer dos tipus nous de paraules clau, un era el join(complement) que havíem d'ignorar per ser informació complementaria. L'altre tipus era un tipus de espai entre gens desconegut. Normalment trobem un sencer amb l'espai que hem de guardar entre els dos gens, però en aquest cas apareixia un espai unk100, que buscant informació per l'NCBI, vaig trobar que significava un espai de gap desconegut, tot i així recomanaven el 100 com a gap arbitrari per fer quadrar l'estructura, així que vaig buscar les cadenes que continguessin el unk100 i les anava substituint per 100 cada cop que apareixien.

El programa fallava  per cadascun d'aquests problemes que he comentat i anaven sorgint L'execució es parava, vol dir que si en un futur el NCBI utilitza un nou tipus de identificador, una nova forma anomenar els cromosomes o descriure les posicions dels gens, el programa fallaria i no es generarien els gens.

És per això que en cada punt on el programa podia fallar vam afegir un control d'errors per poder avisar per exemple a l'administrador i sempre que hi hagin problemes, primerament saber-ho i després tenir informació per solucionar-los.

Per acabar veurem una imatge [Figura 23] on es pot apreciar com coincideix la trobada d'un gran SMUM amb dos gens que són el mateix a cada genoma, Abca12.




[Figura 23] Observem la interfície Mummy i els SMUMs en detall, veiem els gens situats a dalt i abaix dels SMUMs. En aquest cas, veiem com coincideix la trobada d'un gran SMUM i que els dos gens són el mateix. L'Abca12.

## 3.7 Fase 6: Automatització del robot de descarrega de nous genomes eucariotes i de la comparació dels genomes descarregats.

Per la descarrega dels genomes era necessària la connexió amb l'FTP del NCBI [2]. El programa s'havia de realitzar des de cero, ja que per bacteris els genomes es troben en un fitxer comprimit tots junts, i facilita molt la descarrega.

En eucariotes trobem a ftp://ftp.ncbi.nih.gov/genomes/ un seguit de carpetes, alguns de genomes eucariotes, algunes carpetes on hi ha informació de bacteris i es troba tot barrejat i una mica caòtic

Havia de trobar l'ordre, el patró per poder automatitzar la descarrega de genomes, i ho vaig trobar. Cada genoma d'eucariota disponible i que ens serveix per la comparació, és a dir, que podem descarregar els seus fitxers en format FASTA, disposa d'una carpeta que s'anomena Assembled chromosomes. Aquesta carpeta, de vegades conté un fitxer accession, que podem fer servir per la descarrega de gens d'un genoma, així que l'hem de descarregar. I sempre conté un fitxer chr_NC_gi també per la descarrega de gens i també l'hem de descarregar. De vegades els cromosomes que hem de descarregar es troben en aquesta mateixa carpeta, però la majoria de vegades estan a l'interior d'una carpeta anomenada "seq".

Ens podem imaginar el procediment, hem d'anar a prova i error, si existeix aquesta carpeta entrar, si no existeix, buscar aquí, etc.

Per fer tot aquest algorisme vaig decidir utilitzar Java. El programa de mapatge de gens fa servir una llibreria que ens aniria fantàsticament bé, que permet llistar carpetes, navegar per aquestes carpetes, descarregar fitxers i moltes més operacions sobre FTP. La llibreria és la edtftpj, és una llibreria de Enterprisedt [16] que permet el control de sessions FTP en Java.

Així doncs vaig començar iterant la llista(ftp.directoryList(".")) de cada carpeta del FTP del NCBI [2], dintre d'aquesta carpeta iterant per cada carpeta, buscant si contenien una carpeta Assembled_chromosomes. Si la troba entrava descarregava els fitxers pel mapatge de gens i realitzava una comprovació de si existia la carpeta seq, si existia entrava, i si no existia es quedava a la carpeta on estava i realitzava la mateixa sol·licitud de descarrega (ftp.downloadFile) per cada fitxer que compleixi els requisits de cromosoma vàlid, contenir la paraula "_ref_" i acabar en ".fa.gz".

### 3.7.1 No descarregar genomes que ja tenim

Fins aquí ja hem automatitzat la descarrega, però falta polir una serie de detalls. La pròxima vegada que cridi al programa tornarà a descarregar tots els genomes. La manera més senzilla de solucionar això és guardar a un fitxer els noms de les carpetes dels genomes descarregats. Així abans de processar qualsevol carpeta es comprova que no estigui al fitxer, i si hi és, passem a la següent carpeta.

Vaig decidir crear un altre fitxer on escriure manualment les carpetes que no continguin genomes, així ens estalviem de processar-les i guanyem en temps; si hi hagués cap problema amb una carpeta, o no volem descarregar algun genoma, el podem posar en aquest fitxer i el programa no el descarregarà.

### 3.7.2 Actualitzacions de genomes que ja tenim

La genòmica és una ciència que avança a passes de gegant, i només en aquest últims 4 mesos s'han actualitzat gairebé la meitat dels genomes i ha aparegut algun de nou. Així que no podem descarregar els genomes un cop i deixar-ho estar, hem de comprovar la data de modificació si el genoma ja el teníem

Tenim la data de l'última descarrega que hem fet, comprovant amb la classe FILE de Java el fitxer amb el nom de genomes descarregats. I gràcies a la llibreria edtftpj també podem agafar el identificador d'una carpeta i comprova la data d'última modificació ( .lastModified() ). Només hem de comprovar si l'ultima data de modificació del genoma del FTP ha succeït després que la data de modificació del fitxer de genomes descarregat, i descarregar aquell genoma.

### 3.7.3 Identificació de la classificació de l'espècie dins de les eucariotes

Un cop descarregats tots els fitxers que necessitem, hem d'identificar el genoma, ja que es troben tant animals com fongs barrejats. Per això vaig haver d'investigar l'ús de les eines de consulta remota a les bases de dades del NCBI, les e-utils [3]. Les e-utils són un seguit de eines que permeten fer consultes i indicar diverses opcions afegint-les a l'URL des de la que fem la consulta.

Pel nostre cas, havíem de buscar el nom del genoma primerament, per obtenir el ID de la especie dins de la base de dades. Un cop teníem l'ID, podíem consultar el taxò de la especie, això és la classificació que ocupa al regne d'eucariotes, veiem un exemple:

Taxonomy: Eukaryota; Opisthokonta; Metazoa; Eumetazoa; Bilateria; Coelomata; Deuterostomia; Chordata; Craniata; Vertebrata; Gnathostomata; Teleostomi; Euteleostomi; Sarcopterygii; Tetrapoda; Amniota; Mammalia; Theria; Eutheria; Euarchontoglires; Glires; Lagomorpha; Leporidae

Vaig trobar multitud de problemes per buscar l'ID del genoma a la BD "genome"(mitjançant e-utils).Cerca amb la següent adreça:

http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=genome&term=XX
on XX representa el nom del genoma.

Per alguns genomes, funcionava, per altres no trobava cap identificador. Em va costar moltes proves, cerques amb el nom de la carpeta del genoma(pe. D_rerio), separant el nom de la carpeta (D, rerio), vaig fer al programa que descomprimís un cromosoma i agafes de la primera línia el nom del genoma oficial complet (Danio Rerio) i tot i així segons amb el nom que buscava, apareixien els IDs d'algunes especies, o els Ids d'altres especies, però mai totes.

Amb l'ajuda de Mario, vaig descobrir una base de dades nova que no estava publicada al directori de e-utils, la base de dades Taxonomy. En aquesta BD quan buscava el nom del genoma sencer (pe. Danio Rerio) apareixia un únic ID i per tots el genomes funcionava. L'adreça de consulta és la següent:

http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=taxonomy&term=XX
on XX representa el nom del genoma.

El que retorna l'adreça, és un XML que havia de parsejar amb la classe de Java DocumentBuider, i accedir al camp ID.

Amb el valor del ID del genoma de la BD taxonomy podia consultar el seu taxò amb la següent adreça e-útil:

http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?
db=taxonomy&id=XX&retmode=xml
On XX representa la ID que hem obtingut abans.

Parsejant el XML de retorn tenim en un camp tot el taxò de l'espècie i podem organitzar el genoma descarregat a la secció del servidor on calgui.

### 3.7.4 Robot d'automatització bimensual

El robot d'automatització bimensual és un programa que crida a la resta de programes desenvolupats per mantenir les comparacions de genomes eucariotes actualitzades. Veurem al informe tècnic en detall la forma de programar l'execució bimensual del nostre robot d'automatització bimensual. Vaig utilitzar el cron de GNU/Linux per programar l'execució d'aquest robot el primer dissabte de cada dos mesos a la matinada.

La raó d'aquesta programació és primerament perquè volíem fer la comprovació cada dos mesos i després perquè la normativa de les consultes de e-utils:
"...
Frequency, Timing and Registration of E-utility URL Requests
In order not to overload the E-utility servers, NCBI recommends that users post no more than three URL requests per second and limit large jobs to either weekends or between 9:00 PM and 5:00 AM Eastern time during weekdays. Failure to comply with this policy may result in an IP address being blocked from accessing NCBI
..."

Si no complíem aquesta política podíem obtenir un bloqueig de la IP del servidor. A més, guanyaríem velocitat per utilitzar la xarxa i descarregar els genomes en hores de poc treball.

# 4. Informe Tècnic

En aquest apartat explicaré tots els processos, programes, carpetes i fitxers que intervenen a l'automatització de comparació de genomes eucariotes. La intenció és reproduir el procediment a baix nivell perquè altres desenvolupadors puguin continuar amb la línia de treball.

Primerament, esmentaré tota l'estructura de carpetes i arxius del servidor http://platypus.uab.cat [4] perquè el lector pugui seguir sense problemes l'explicació dels programes.

Seguidament, per l'explicació de tot el software desenvolupat i que quedi clar tot el procediment que segueix l'automatització de comparació de genomes, seguiré un estricte ordre d'execució. D'aquesta manera veurem els programes de més abstracció a menys, és a dir, primer veurem el robot d'automatització bimensual i conforme va cridant la resta de programes anirem baixant. Si en la explicació d'un programa es fa la crida a un altre programa, s'indicara que la explicació de aquest és trobarà al següent punt.

## 4.1 Estructura d'arxius i carpetes del software generat

Organitzaré l'estructura de carpetes en taules per poder explicar cada fitxer. Com que no em caben totes les carpetes indicaré amb un asterisc (pe. *carpeta/) les carpetes que explicaré a part. Comencem per la carpeta arrel del software, la podem trobar al directori "/var/www/cgi-bin/genomas/RobotEucariota/".

| Carpeta | Fitxers i carpetes | Explicació |
|---|---|---|
| RobotEucariota/ | makefile | Per compilar el RobotEucariota.cc . |
| | RobotEucariota.cc | Programa d'automatització bimensual. |
| | *downloadRobot/ | Carpeta que conté el programa de descarrega de genomes eucariota i tot el necessari per l'execució. |
| | *sync_genes/ | Carpeta que conté el programa de mapatge de gens sobre el genoma i tot el necessari per l'execució. |
| | *SinRenombrar/ | Aquesta carpeta conté diversos programes del procés post-descarrega. |
| | *actualitza_mums/ | Aquí disposem de tots els programes on es generen les comparacions. |

| Carpeta | Fitxers i carpetes | Explicació |
|---|---|---|
| downloadRobot/ | DownloadRobot.jar | Programa de descarrega de genomes eucariota. |
| | genesdescargados.txt | Fitxer amb una llista de noms dels genomes eucariotes que ja tenim descarregats al nostre servidor [4]. |
| | no.txt | Fitxer amb noms de carpetes conegudes del FTP que el programa ha d'ignorar per no contenir genomes. |
| | nuevosMetazoa.txt | Fitxer que genera el programa amb una llista dels genomes que ha descarregat durant l'execució. |
| | output.txt | Guardem la sortida del programa per si hem de consultar alguna cosa. |
| | bin/ | Fitxers de classes. |
| | genome/ | En aquesta carpeta es descarreguen els nous genomes. Per cada genoma es crea una carpeta amb el nom del genoma. És una carpeta d'ús temporal. |

| Carpeta | Fitxers i carpetes | Explicació |
|---|---|---|
| sync_genes/ | getgenes.jar | Programa de mapatge de gens al genoma. |
| | genes.txt | Aquest és un fitxer es necessari ja que conté una llista amb tots els genomes dels que el programa ha de descarregar els gens pel mapatge. |
| | Assemblies/ | En aquesta de carpeta, hem de crear una carpeta per cada genoma que vulguem que descarregui els gens. Aquesta carpeta ha de tenir el nom del genoma i ha de contenir el fitxer Accession.txt del genoma. Explicarem aquest fitxer en detall quan expliquem el getgenes.jar |
| | bin/ | Fitxers de classes i fitxer de configuració de carpetes: local del programa i la remota del servidor FTP NCBI[2]. |
| | lib/ | Llibreries que utilitza el programa. |
| | /genomas/Eukaryota/ mapedgenes/ | Dins d'aquestes carpetes es generen el fitxers de gens nºID.gen. Després es mouen a la carpeta del genoma per ser renombrats per l'identificador del genoma. |

| Carpeta | Fitxers i carpetes | Explicació |
|---|---|---|
| SinRenombrar/ | compilar.sh | Script per compilar els tres programes d'aquesta carpeta. |
| | ngenoma | Fitxer que conté un numero que representa la quantitat de genomes al servidor. S'incrementa cada cop que descarreguem un genoma i serveix per obtenir el identificador únic. |
| | procesoDescarga.cc | Programa que gestiona la inserció del genoma al sistema a la carpeta que toca, amb el format i nom que han d'anar els fitxers i carpetes. |
| | renombrar.cc | Programa que s'encarrega de canviar els noms dels genomes descarregats per el numero identificador. |
| | unirCromosomas.cc | Programa que genera el fitxer de genoma complet, a base de concatenar en ordre els cromosomes. |
| | procesados/ | Carpeta que conté tots els genomes processats, es fa una copia de seguretat per si el procés d'inserció al sistema és erroni.<br>Nota important: És l'única carpeta que pot haver, tret de les carpetes de genomes a processar. |

| Carpeta | Fitxers i carpetes | Explicació |
|---|---|---|
| actualitza_mums/ | offsetscromo | Fitxer que conté el desplaçament de cada cromosoma, necessari per fer la concatenació de MUMs de cromosomes correctament. El genera el MUMOL durant la comparació. |
| | logprocessats | Fitxer que guarda els identificadors de les parelles de genomes que s'han comparat correctament. |
| | make | Fitxer per compilar el programa actualitza_mums. |
| | actualitza_mums.cc | Programa per llançar la comparació de genomes actualitzats o nous. |
| | lanza_smums.cc | Programa que llança el càlcul de SMUMs de una comparació. |
| | separasmums.sh | Script per eliminar duplicats de SMUMs a la ultima volta del càlcul de SMUMs. |

| | | |
|---|---|---|
| | smumsortV3.sh | Script que transforma el fitxer de SMUMs per poder ser llegit pel Mummy. |
| | libera_sem.sh | Script encarregat de eliminar els semàfors de sistema declarats i que no es fan servir. |
| | mums/ | Carpeta que allotja tots els MUMs de cromosomes. |
| | smums/ | Carpeta temporal on van a parar els smums durant la seva generació. Després es traslladen a les carpetes següents. Els SMUMs de la tercera passada es guarden en aquesta carpeta fins l'execució dels scripts separasmums i smumsortV3. Els resultats es traslladen a la carpeta del directori superior, smumsort/ . |
| | smums_paso1/ | Carpeta on es guarden els SMUMs resultants de la primera passada. |
| | smums_paso2/ | Carpeta on es guarden els SMUMs resultants de la segona passada. |
| | smums_paso3/ | Carpeta on es guarden els SMUMs resultants de la tercera passada. |
| | smumsort/ | Carpeta d'us temporal. |
| | factors/ | Carpeta on es guarden els fitxers temporals per la generació del fitxer factors.txt |
| | long_Mumunix/ | Aquesta carpeta conté els programes i llibreries per executar el algorisme MUMOL pel càlcul de MUMs. |
| | calculoSuperMums/ | Aquesta carpeta conté els programes i llibreries per executar el programa smum.cc pel calcul de SMUMs. |
| | Concatena/ | Conté el programa necessari per la concatenació de MUMs de cromosomes. |
| | EliminaNoMum/ | Conté el programa necessari per eliminar els no-MUMs i el programa per eliminar SMUMs duplicats. |

Seguidament veurem l'estructura de la carpeta animalia/ , aquesta carpeta conté els genomes eucariotes dels animals i totes les dades resultants de les comparacions. La podem trobar a:

"/var/www/cgi-bin/genomas/Eukaryota/animalia/"

| Carpeta | Fitxers i carpetes | Explicació |
|---|---|---|
| animalia/ | factors.txt | Aquest fitxer conté una línia per cada comparació i apareix el valor de semblança dels dos genomes. Es genera amb el programa uneFactors.cc i s'utilitza per crear l'arbre de distàncies de genomes. |
| | genes.txt | Aquest fitxer conté el nom de tots els genomes que disposem al nostre servidor [4]. L'ordre d'aquest fitxer és importantíssim doncs la posició que ocupa el nom en aquest fitxer és l'identificador únic del genoma. Fa de traducció entre el nom del genoma i la seva ID. |
| | genome/ | Conté tots els genomes dels que disposem al nostre servidor [4]. Per cada genoma conté una carpeta que com a nom té el identificador únic del genoma. Dintre de la carpeta conté tots els cromosomes, el genoma complet i una carpeta "info/" amb altres fitxers del genoma, accession.txt, entre altres).<br>Nota important: En aquesta carpeta, i en les descendents no poden haver més fitxers ni carpetes fora dels esmentats. |
| | old/ | Quan s'actualitza un genoma, guardem el genoma antic per seguretat. |
| | mapedgenes/ | Aquesta carpeta conté els gens generats pel programa de mapatge de gens. Es guarden com nºID.gen. Aquests fitxers els carrega l'aplicatiu web Mummy per visualitzar la comparació. |
| | mums/ | Carpeta que conté els MUMs finals de les comparacions de tots els genomes eucariotes. |
| | smums/ | Carpeta que conté els SMUMs finals de les comparacions de tots els genomes eucariotes. |
| | smumsort/ | Carpeta que conté els SMUMs de les comparacions amb l'index d'ordenació. Aquests fitxers són els que pot carregar l'aplicatiu web Mummy per visualitzar la comparació. |

## 4.2 Robot d'automatització bimensual

El robot d'automatització bimensual no deixa de ser un programa en C++ que crida a la resta de programes desenvolupats per mantenir les comparacions de genomes eucariotes actualitzades.

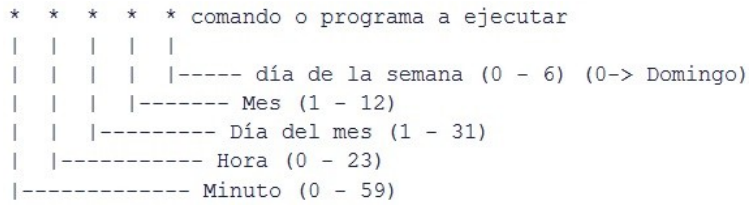
La forma de programar l'execució bimensual del nostre robot d'automatització bimensual ha de ser amb eines del sistema, en aquest cas GNU/Linux, així que hem de fer servir el cron.

Les comandes d'ús són les següents:

#crontab -l , per llistar el contingut del fitxer cron, és a dir, les planificacions programades al sistema.

#crontab -e , per editar el fitxer cron i modificar planificacions programades i programar-ne de noves. S'utilitza l'editor vi.

El funcionament del fitxer cron ve explicat a la següent figura [Figura 24]




*Img. 4.2.1: Configuración del cron*

[Figura 24] Funcionament del fitxer cron per programar tasques.

He programat la següent línia per executar el software el primer dissabte cada dos mesos a les 2:15 de la matinada.

15 2 1,2,3,4,5,6,7 1,3,5,7,9,11 6 /var/www/cgi-bin/genomas/Robot

El motiu és senzill, s'explicara en detall a la secció del Robot de descarrega i a la secció del programa de mapatge de gens, però per fer-nos una idea: les connexions amb l'FTP del NCBI i les grans consultes a les bases de dades s'han de fer en horari nocturn els caps de setmana. D'aquesta manera també alliberem carrega a la UAB i al mateix IBB assegurant-nos que no molestarem a ningú en aquest horari.

**robot.cc :**

**Descripció general:**

Al inici comprova si ha acabat l'ultima actualització (llançada al anterior timming), si encara està executant-se, el programa es tanca. Si l'execució de la comparació anterior va acabar crida successivament a diversos programes. Primerament crida al **robot de descarrega de nous genomes i seqüències de genomes actualitzades** (s'explicarà al següent apartat, 5.2.1). Es comprova si aquest programa ha fet alguna descarrega nova. Si no hi han novetats el programa es tanca. Si hi han novetats cridem al **programa de descarrega del mapatge de gens** (s'explicarà als següents apartats, 5.2.2).

Un cop descarregats els gens i el genoma, cridem al **programa d'inserció de nous genomes al servidor per ser comparats** (s'explicarà als següents apartats, 5.2.3) per inserir el genoma al nostre sistema de forma adequada.

Després de la inserció del genoma, generem una llista amb el numero identificador dels nous genomes que hem de comparar contra tota la resta de genomes.

Passem la llista de genomes a comparar al **programa actualitza_mums perquè realitzi la incorporació de nous genomes a la comparació tots amb tots** (s'explicarà als següents apartats, 5.2.4).

Una vegada acabada la comparació dels nous genomes amb la resta de genomes acaba el procés.

**Procediment de compilació:**

Disposem de dos opcions:
-Executant la comanda make al directori Robot.
-Cridant al compilador: g++ robot.cc -o Robot

**Procediment d'execució:**

Tot i que la execució és cada dos mesos, podem executar el robot amb la següent comanda: ./Robot

**Dades d'entrada:**

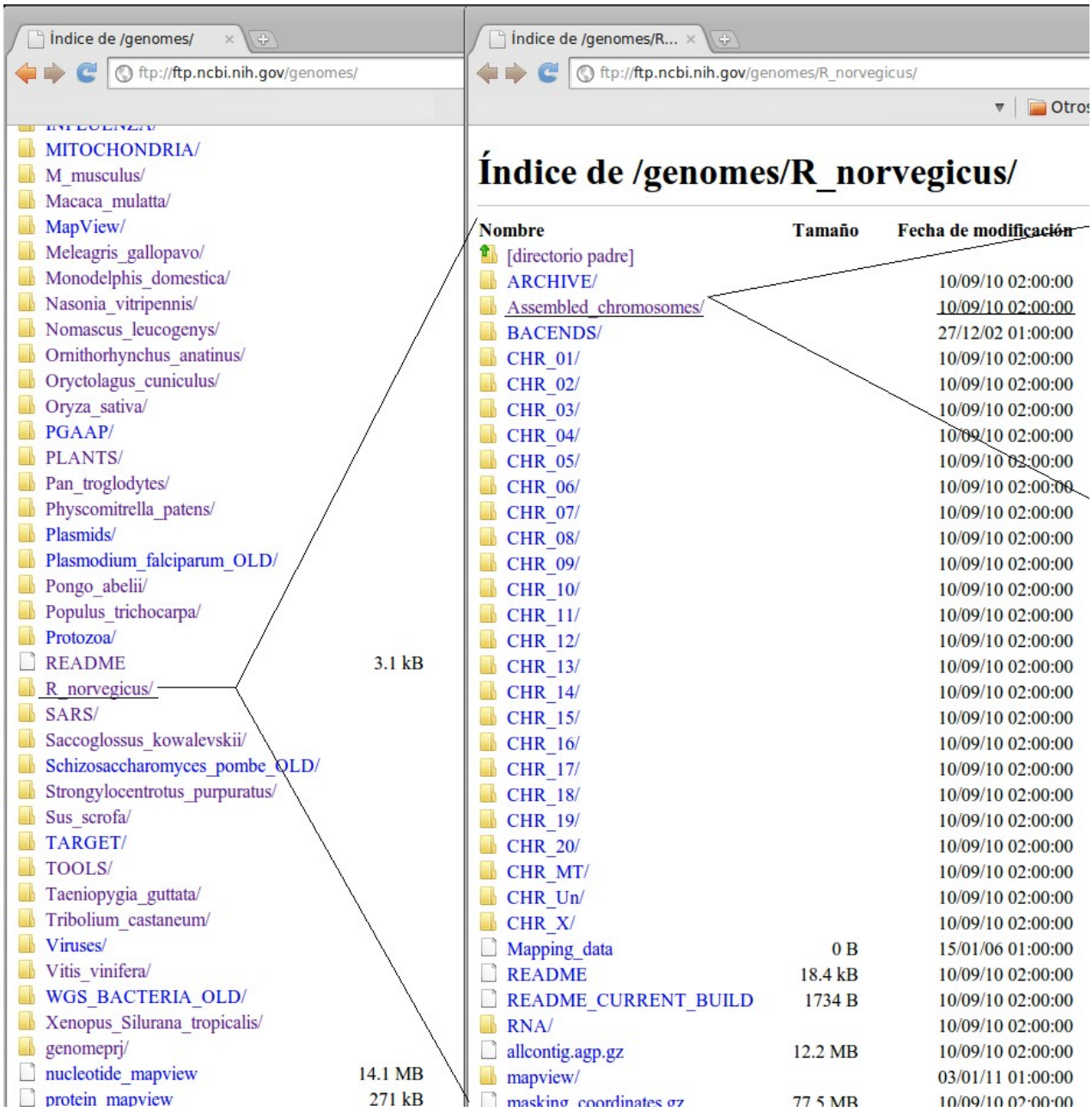
No necessita cap paràmetre per part de l'usuari.

**Dades de sortida:**

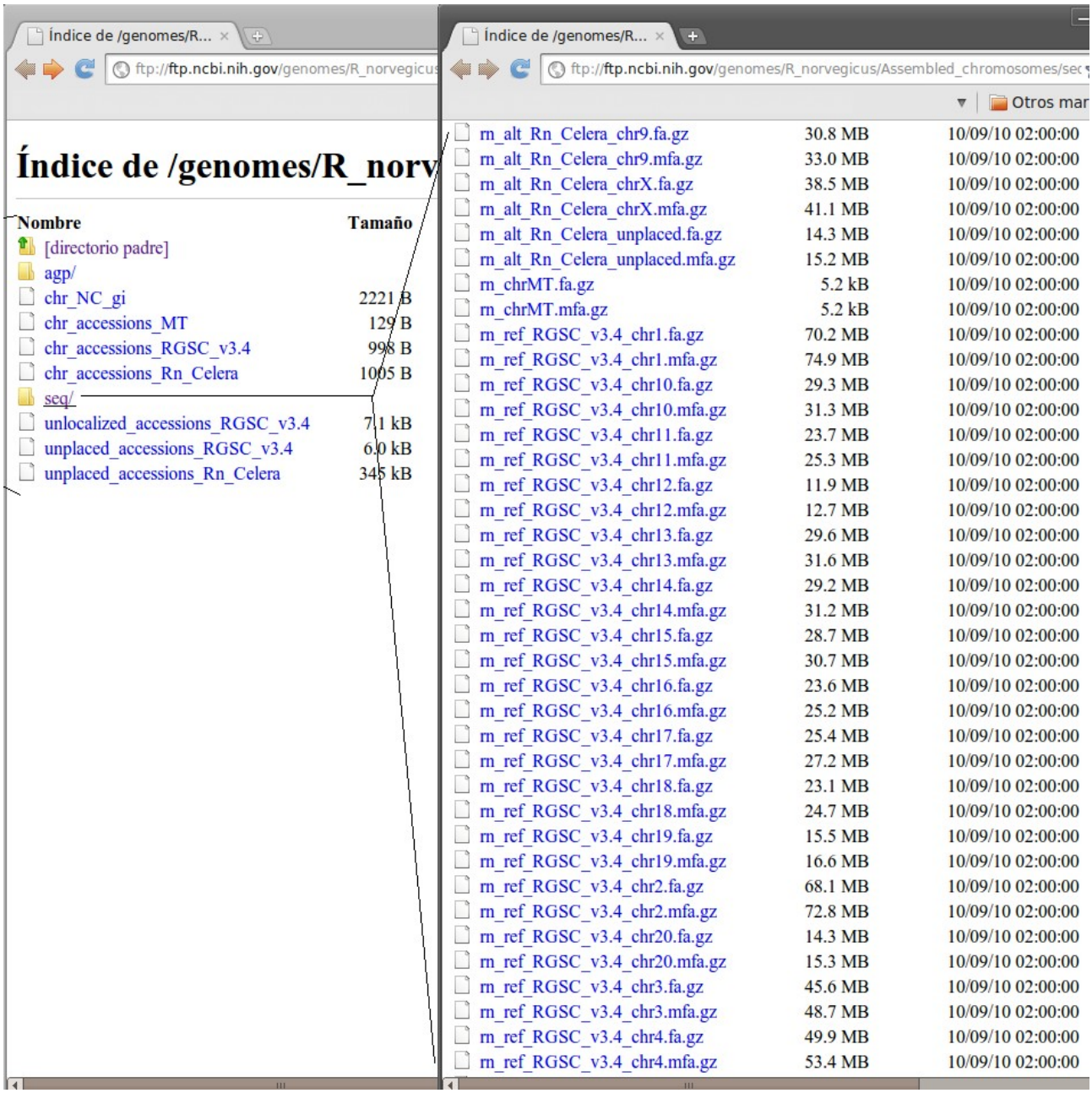
No genera cap sortida més que la crida a altres programes i les dades que deriva cap aquests.

### 4.2.1 Robot de descarrega de nous genomes i seqüències de genomes actualitzades

El robot de descarrega de nous genomes i seqüències de genomes actualitzades ha de navegar per l'FTP del NCBI i comprovar si han hagut novetats de genomes eucariotes. Si hi han novetats descarregar-les. Per entendre el funcionament del programa, veiem l'estructura de carpetes del FTP del NCBI [Figures 25 i 26]. Tots els genomes eucariota de qualsevol classificació es troba en la adreça ftp://ftp.ncbi.nih.gov/genomes/ .




[Figura 25] A l'esquerra de la imatge veiem la carpeta general de genomes eucariotes, com exemple el genoma de la rata es troba a la carpeta R_norvegicus/. A la dreta veiem el contingut de d'aquesta carpeta. Existeix informació mitocondrial, de gens, versions anteriors i dins la carpeta Assembled_chromosomes trobem el genoma.




[Figura 26] A l'esquerra de la imatge veiem el contingut de la carpeta Assembled_chromosomes/ segons l'estat de la seqüenciació del genoma podem trobar els fitxers en aquesta carpeta o bé a l'interior de la carpeta seq/. Observem els fitxers accessions necessaris pel mapatge de gens. A la dreta observem els fitxers dels cromosomes i diferents versions del genoma, alternatives i referencials.

**DownloadRobot.jar :**

**Descripció general:**

El robot de descarrega de genomes eucariota primerament carrega en memòria els dos fitxers, genesdescargados.txt i no.txt, busca a la carpeta FTP del NCBI (o la indicada al fitxer config.properties) i comprova carpeta per carpeta.

Si el nom de la carpeta està inclòs a qualsevol dels dos fitxers esmentats, vol dir que o bé es tracta d'una carpeta que no conté genomes o bé que ja disposem del genoma. Si es tracta de l'últim cas comprovem la data de modificació del genoma i només si es posterior a la data de la nostra versió el descarreguem.

Per la resta de carpetes comprovem si es tracten de genomes mirant l'interior. Si contenen una carpeta Assembled_chromosomes/ no només és un genoma que ens interessa, sinó que està seqüenciat, almenys part.

Un cop descarregats tots els fitxers que necessitem, hem d'identificar el genoma, ja que es troben tant plantes com peixos barrejats. Per això fem us de les eines de consulta remota a les bases de dades del NCBI, les e-utils [3]. Després d'obtenir el taxò de l'espècie (la classificació dins d'eucariotes) escrivim si es tracta d'un animal (Metazoa) dins del fitxer nuevosMetazoa.txt .

Després d'això el programa finalitza el procés.

**Procediment de compilació:**

-Per compilar fàcilment, podem fer us de l'entorn Eclipse [15].

-Creem un nou projecte indicant la carpeta del codi del programa. L'Eclipse detecta automàticament el directori de libs, src, etc.

-Compilar perquè generi tot el necessari.

-Anem a File -> Export -> Runnable JAR file -> Export destination i verifiquem que tenim marcada l'opció "package required libs in jar", així empaquetem tot el necessari dins del jar.

-Abans d'executar, comprovem que l'arxiu de configuració bin/config.properties conté els paths correctes on es troba l'aplicació.

-Ja el tenim preparat per executar.

**Procediment d'execució:**

Indiquem que executem un jar, les opcions en memòria i que ens guardi la sortida del programa al fitxer output.txt:

java -Xmx512m -jar DownloadRobot.jar > output.txt

**Dades d'entrada:**

**-**Necessita el fitxer bin/config.properties ben configurat, exemple:

remote_host=**ftp.ncbi.nlm.nih.gov** (indiquem ftp remot)
local_directory_Eukaryota=/path_on_trobem_programa/downloadRobot/
remote_directory_Eukaryota=/genomes/ (indiquem carpeta d'eucariotes)

-genesdescargados.txt : Fitxer amb una llista de noms dels genomes eucariotes que ja tenim descarregats al nostre servidor. Si esta buida, el programa descarregarà tot el que trobi.

-no.txt : Fitxer amb noms de carpetes conegudes del FTP que el programa ha d'ignorar per no contenir genomes.

**Llibreries utilitzades:**

-edtftpj: edtftpj és una llibreria de Enterprisedt [16] que permet el control de sessions FTP en Java.

**Dades de sortida:**

-genesdescargados.txt : Per cada descarrega, el programa afegeix una línia amb el nou genoma. Si el fitxer és una actualització, no afegeix res.

-nuevosMetazoa.txt : Per cada descarrega del regne animal, el programa afegeix una línia, tant si és una nova descarrega, com si és una actualització.

-/genomas/Eukaryota/X/ : Genera tantes carpetes X, amb el nom del genoma, per cada genoma nou o actualitzat que descarrega. A l'interior descarrega el genoma.

### 4.2.2 Programa de mapatge de gens sobre el genoma

Programa encarregat de generar el mapatge de gens sobre el genoma. Descarrega els fitxers amb la informació genètica de cada cromosoma i situa cada gen a la seva posició corresponent dins el genoma.

**getgenes.jar :**

**Descripció general:**

Com hem vist pàgines enrere a la [Figura 26], disposem a la adreça ftp://ftp.ncbi.nih.gov/genomes/ les carpetes dels genomes. Dintre d'aquestes carpetes, on es troba la carpeta Assembled_chromosomes/ existeixen altres carpetes amb el nom dels cromosomes. Aquestes carpetes contenen diversos fitxers, gbs, gbk amb informació del cromosoma. Podem processar els fitxers gbs per obtenir els gens.

El programa recorre aquestes carpetes pels genomes indicats a genes.txt i descarrega els fitxers necessaris. Per cada fitxer gbs descarregat i per cada entrada de cromosoma al fitxer accession, el programa accedeix i descarrega la informació d'ensamblatge dels gens i processa tota la informació per generar els fitxers .gen.

Després de generar els fitxer de gens el programa finalitza l'execució.

**Procediment de compilació:**

-Per compilar fàcilment, podem fer us de l'entorn Eclipse [15].

-Creem un nou projecte indicant la carpeta del codi del programa. L'Eclipse detecta automàticament el directori de libs, src, etc.

-Compilar perquè generi tot el necessari.

-Anem a File -> Export -> Runnable JAR file -> Export destination i verifiquem que tenim marcada l'opció "package required libs in jar", així empaquetem tot el necessari dins del jar.

-Abans d'executar, comprovem que l'arxiu de configuració bin/config.properties conté els paths correctes on es troba l'aplicació.

-Ja el tenim preparat per executar.

**Procediment d'execució:**

Indiquem que executem un jar, les opcions en memòria i que ens guardi la sortida del programa al fitxer output.txt:

java -Xmx1024m -jar getgenes.jar -type=[Archaea|Bacteria|Eukaryota] -downloadfiles=[true|false] > output.txt

**Dades d'entrada:**

**-**Necessita el fitxer bin/config.properties ben configurat, exemple:

remote_host=**ftp.ncbi.nlm.nih.gov** (indiquem ftp remot)
local_directory_Eukaryota=/path_on_trobem_programa/sync_genes/
remote_directory_Eukaryota=/genomes/ (indiquem carpeta d'eucariotes)

-El paràmetre *type* determina el domini al que pertanyen els genomes.

-El paràmetre *downloadfiles* determina si es necessària la descarrega dels fitxers per la generació dels gens.

-El fitxer genes.txt és un fitxer necessari ja que conté una llista amb tots els genomes dels que el programa ha de descarregar els gens pel mapatge. El genera el downloadrobot.jar i el robot d'automatització s'encarrega de col·locar-lo al seu lloc.

-Per cada genoma que vulguem descarregar els gens ha d'existir un subdirectori de la aplicació amb el format: /assemblies/X/ on X és el nom del genoma que descarregarem. Aquesta carpeta ha de contenir un fitxer Accession.txt amb el següent format:

<Identificador de cromosoma> <RefSeq><Accession.version>

Tant RefSeq com Accession.version són diferents identificadors a les bases de dades del NCBI dels fitxers GenBank corresponents a cada cromosoma.
Aquest fitxer el descarrega el downloadRobot.jar i el robot d'automatització s'encarrega de col·locar-lo al seu lloc.

**Llibreries utilitzades:**

-edtftpj: edtftpj és una llibreria de Enterprisedt [16] que permet el control de sessions FTP en Java.

-BioJava: BioJava és un conjunt de libreries de java per el procés de dades biològiques. Al programa en qüestió el fem servir per llegir els fitxers gbs.

**Dades de sortida:**

-A més de generar, i descarregar dades temporals que eliminem després de l'execució, crea els fitxers de gens de cada genoma a:
/genomas/Eukaryota/mapedgenes/X.gen on X és l'ID del genoma

### 4.2.3 Programa d'inserció de nous genomes al servidor per ser comparats

Programa que gestiona la inserció del genoma al sistema a la carpeta que toca, amb el format i nom que han d'anar els fitxers i carpetes.

**procesoDescarga.cc :**

**Descripció general:**

El programa recorre totes les carpetes del directori, ignorant la carpeta processats, i per cada una (per cada genoma) realitza el següent procediment:

Descomprimeix els fitxers del genoma.

Mou la informació de genomes que no necessitem a la carpeta info del genoma.

Busca si el genoma existia al sistema per aprofitar la ID, o obté una nova.

Copia el genoma a la carpeta processats i crida al programa per **renombrar tots els fitxers del genoma amb l'identificador del genoma** i del programa de **concatenació de la seqüència dels cromosomes d'un genoma** (els dos programes s'explicaran als següents apartats, 5.2.3.1 i 5.2.3.2 ).

Mou el genoma correctament modificat a la carpeta que pertoca i els gens a la carpeta de ../mapedgenes/

El programa finalitza.

**Procediment de compilació:**

Disposem de dos opcions:

-Executant la comanda sh compilar.sh

-Cridant al compilador: g++ procesoDescarga.cc -o procesoDescarga

**Procediment d'execució:**

./procesoDescarga

**Dades d'entrada:**

No necessita cap paràmetre per part de l'usuari.

-A la carpeta SinRenombrar/ només han d'existir les carpetes dels genomes a processar amb el nom del genoma tret de processats/.

-ngenoma : Llegeix el fitxer que conté el numero que representa la quantitat de genomes al **servidor**, per otorgar nous identificadors únics.

**Dades de sortida:**

-processats/X/ : Es fa una copia dels genomes abans de processar-los.

-../genome/X/ : Col·loca el genoma al seu lloc per poder fer les comparacions.

-../mapedgenes/X.gen : Col·loca els gens al seu lloc.

-../genes.txt : Es modifica/afegeix si cal, els nous genomes al fitxer genes.txt de la carpeta superior. S'ha d'afegir en ordre ja que la posició que ocupa el genoma al fitxer determina el número identificador que té el genoma al sistema.

-ngenoma : Actualitza el fitxer que conté un numero que representa la quantitat de genomes al servidor.

#### 4.2.3.1 Programa per renombrar els fitxers de genomes amb l'identificador del genoma

Programa que s'encarrega de canviar els noms dels genomes descarregats per el seu número identificador.

**renombrar.cc :**

**Descripció general:**

El programa recorre tots els fitxers del directori, ignorant la carpeta info, i per cada fitxer (cromosoma)realitza el següent procediment:

Obté la part identificativa del cromosoma del nom.

Utilitzant expressions regulars, trobem quin tipus de cromosoma estem tractant, si és numèric, lletres, cromosomes especials X,Y,Z,W, LG,LGE etc.

Finalment renombra el fitxer amb el format adequat:

<numID_genoma>-<IdCromosoma>[<num_IdCromosoma>]

Fi del programa.

**Procediment de compilació:**

Disposem de dos opcions:

-Executant la comanda sh compilar.sh

-Cridant al compilador: g++ renombrar.cc -o renombrar

**Procediment d'execució:**

./renombrar <Carpeta_genoma/> <num_per_renombrar>

**Dades d'entrada:**

-A la carpeta SinRenombrar/ han d'existir les carpetes dels genomes a processar amb el nom del genoma.

- El paràmetre c*arpeta_genoma/* indica la carpeta on és el genoma a renombrar.

- El paràmetre *num_per_renombrar* indica el numero identificador pel qual s'ha de substituir el nom del genoma.

**Dades de sortida:**

-Fitxers del genoma i de gens renombrats.

### 4.2.3.2 Programa de concatenació de la seqüència dels cromosomes d'un genoma

Programa que genera el fitxer de genoma complet, a base de concatenar en ordre els cromosomes.

**unircromosomas.cc :**

**Descripció general:**

El programa recorre tots els fitxers del directori, ignorant la carpeta info, i per cada fitxer (cromosoma)realitza el següent procediment:

Obté la part identificativa del cromosoma del nom.

Trobem quin tipus de cromosoma estem tractant, si és numèric, lletres, cromosomes especials X,Y,Z,W, LG, etc. i els anem afegint en ordre en vectors de memòria dinàmica. Suposem que en un genoma que disposa de cromosomes X,Y no disposa de cromosomes W,Z i viceversa.

Finalment concatenem els fitxer de cromosomes en ordre seguint els vectors creats, el fitxer final generat té el següent tipus:

<numID_genoma>

Després finalitza l'execució.

**Procediment de compilació:**

Disposem de dos opcions:

-Executant la comanda sh compilar.sh

-Cridant al compilador: g++ unirCromosomas.cc -o unirCromosomas

**Procediment d'execució:**

./unirCromosomas <Carpeta_genoma/> <num_genomar>

**Dades d'entrada:**

-A la carpeta SinRenombrar/ han d'existir les carpetes dels genomes a processar amb el nom del genoma i els cromosomes de l'interior han de estar renombrats.

- El paràmetre c*arpeta_genoma/* indica la carpeta on és el genoma a renombrar.

- El paràmetre *num_per_renombrar* indica el numero identificador pel qual s'han renombrat els noms dels cromosomes del genoma.

**Dades de sortida:**

- Fitxer de genoma complet a base de concatenar cromosomes en ordre.

### 4.2.4 Programa d'incorporació de nous genomes seqüenciats o seqüències de genomes actualitzades a la comparació de genomes tots amb tots

Programa que automatitza tot el procés de comparació i llança la comparació de genomes actualitzats o nous contra la resta de genomes del nostre servidor http://platypus.uab.cat [4].

**actualiza_mums.cc :**

**Descripció general:**

Inicialment el programa crea les carpetes necessàries i processa la cadena de caràcters que rep per paràmetre Genera un vector amb els genomes a comparar i segueix dos bucles que li permeten comparar els genomes del vector entre ells sense repetir comparacions i també comparar-los contra la resta de genomes.

Per cada comparació es comprova que existeixin els genomes i es comprova la mida que ocupa cadascun. Comparem sempre el genoma amb cromosomes més petits, partits en cromosomes. D'aquesta manera tenim suficient memòria RAM per les comparacions.

Creem vectors dinàmics ordenant els cromosomes per mida, guardant el nom del cromosoma i la mida que ocuparà la comparació en memòria RAM. Així podem controlar l'error abans que succeeixi. A partir d'aquí comença la paral·lelització i fem tantes crides al **MUMOL** (l'explicarem al següent apartat, 5.2.4.1) com ens permeti la memòria.

Un cop generats tots els MUMs de cromosomes, cridem al programa de **concatenació de MUMs de la comparació per cromosomes** (l'explicarem als següents apartats, 5.2.4.2) i seguidament cridem al programa de **eliminació de no-MUMs** (l'explicarem als següents apartats, 5.2.4.3)

Després de la generació de MUMs tornem a paral·lelitzar, permetent que el programa principal continuï amb la següent comparació i mentrestant es fa el càlcul de SMUMs cridant al programa **de generació de SMUMs a partir dels MUMs** (l'explicarem als següents apartats, 5.2.4.4). Es realitzen tres passades del càlcul de SMUMs per aconseguir SMUMs més grans. Entre cada passada es crida al programa de **correcció de SMUMs duplicats** (l'explicarem als següents apartats, 5.2.4.5).

Quan totes les comparacions han acabat, es crida al programa de **generació del fitxer de similitud entre genomes tots per tots** i l'script smumsort per **optimitzar els SMUMs per ser llegits per l'applet Mummy** (Tots dos explicats als apartats, 5.2.4.6 i 5.2.4.7), després el programa acaba d'executar-se.

**Procediment de compilació:**

Disposem de dos opcions:

-Executant la comanda make

-Cridant al compilador: g++ actualiza_mums.cc -o actualiza_mums

**Procediment d'execució:**

./actualiza_mums <carpeta_genomes/> <genomes_a_comparar>

**Dades d'entrada:**

- El paràmetre c*arpeta_genoma/* indica la carpeta on són els genomes a comparar.

- El paràmetre *genomes_a_comparar* ha de ser una cadena de caràcters amb els genomes nous o actualitzats a comparar separats per comes. La cadena es del tipus "2,5,7,10".

**Dades de sortida:**

-logprocessats - Fitxer que guarda els identificadors de les parelles de genomes que s'han comparat correctament.

-Fa diverses crides a tots el programes necessaris per la generació de MUMs, SMUMs, i la reste de dades de la comparació. S'encarrega de processar-les i col·locar-les al seu lloc.

### 4.2.4.1 Programa de generació de MUMs, el MUMOL

Programa que em van facilitar al IBB per executar l'algorisme MUMOL pel càlcul de MUMs. Aquest programa executa la comparació de genomes fitxer contra fitxer. Per fer la comparació cromosoma-genoma, el programa actualitza_mums s'encarrega de tot per habilitar aquesta comparació.

**mumol.cc :**

**Descripció general:**

Aquest programa troba els MUMs de dos fitxers, vam haver d'afegir diverses adaptacions per habilitar la comparació a eucariotes:

-Renombrament de fitxers per no solapar fitxers a la paral·lelització.

-Detectar i escriure els fitxers de desplaçaments de les zones no seqüenciades. (pe. ACTGTNNNNNNNTGA)

-Escriure el fitxer de offsetscromo per permetre la concatenació dels MUMs de cada cromosoma sumant els desplaçaments adients a cada cromosoma.

-Afegir l'ús de semàfors per no escriure paral·lelament a un fitxer.

**Procediment de compilació:**

-Cridant al compilador: g++ mumol.cc -o mumol

**Procediment d'execució:**

./mumol <fitxer1> <fitxer2> <nom_evitar_solapacions> <nom_cromosoma> <minimum_matching> <bool_inverse>

**Dades d'entrada:**

-El paràmetre *fitxer1* inclou el path del primer fitxer a comparar.

-El paràmetre *fitxer2* inclou el path del segon fitxer a comparar.

-El paràmetre *nom_evitar_solapacions* inclou una cadena de caràcters per afegir als noms que puguin solaparse per la paral·lelització. Aquesta cadena inclou els identificadors dels genomes i del cromosoma.

-El paràmetre *nom_cromosoma* porta el nom del cromosoma per escriure'l als fitxers que ho necessitin.

-El paràmetre *minimum_matching v*alor que indica la mínima coincidència que ha de trobar per considerar una seqüència com a MUM.

-El paràmetre bool_inverse indica si la cerca de MUMs és directa (0) i inversa (1).

**Dades de sortida:**

-Fitxers de sortida de MUMs.

-Fitxers offset1 i offset2 per la correcció de les seqüències incompletes

-Fitxer offsetsCromo per la concatenació de MUMs de cromosomes.

### 4.2.4.2 Programa de concatenació de MUMs de la comparació per cromosomes

Programa que genera el fitxer de MUMs del genoma, a base de concatenar en ordre les comparacions de cada cromosoma.

**concatena.cc :**

**Descripció general:**

El programa ordena i llegeix el fitxer chr_offsets. Genera un vector dinàmic ordenat per anar guardant els cromosomes especials amb lletres, LG, etc. Quan té tot ordenat passa a concatenar els MUMs de cromosomes, afegint el desplaçament necessari als valors de posició de cada cromosoma.

**Procediment de compilació:**

-Cridant al compilador: g++ concatena.cc -o concatena

**Procediment d'execució:**

./Concatena/concatena <num_genoma_particionat> <num_genoma_sense_partir>

Nota: S'ha d'executar des de la carpeta superior.

**Dades d'entrada:**

- El paràmetre *num_genoma_particionat* indica l'identificador del genoma que s'ha comparat per cromosomes.
- El paràmetre *num_genoma_sense_partir* indica l'identificador del genoma que s'ha comparat sencer.
- ../../mums/offsetscromo/<num_genoma_particionat>/chr_offsets – Aquest fitxer ha d'existir, el genera el MUMOL i escriu els desplaçaments que necessita cada cromosoma.

**Dades de sortida:**

- Fitxer MUM de la comparació a la carpeta mums/

| | **Fitxer exemple de MUMs** | | |
|---|---|---|---|
| MUM 1 | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del MUM |
| MUM 2 | Posició inicial genoma 2 | Posició inicial genoma 2 | Mida del MUM |
| | … | … | … |
| MUM N | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del MUM |

### 4.2.4.3 Programa de eliminació de no-MUMs

Programa necessari per eliminar els no-MUMs generats per fer la comparació per cromosomes.

**corregirNoMum.cc :**

**Descripció general:**

El programa ordena el fitxer de MUMs per la segona columna que és la problemàtica perquè poden caure més d'un MUM de diferents cromosomes.

A partir d'aquí recorre en ordre la segona columna i va generant un vector dinàmic on va guardant les posicions de cada MUM que pot absorbir altres MUMs més petits dintre seu. Així va avançant la lectura del fitxer i escrivint els MUMs correctes en un altre fitxer.

Els MUMs han de tenir un format com aquest:

<num_petit_genoma>_<num_gran_genoma>.[directo|inverso]

Si el genoma per raons de la comparació per cromosomes te el tipus 6_3.directo, o 20_2.inverso, s'han d'invertir per 3_6.directo i 2_20.inverso, invertint també les columnes perquè el programa Mummy pugui processar les dades. Aquest programa també corregeix això.

**Procediment de compilació:**

-Cridant al compilador: g++ corregirNoMum.cc -o corregirNoMum

**Procediment d'execució:**

./corregirNoMum <Fitxer_MUMs>

**Dades d'entrada:**

- El paràmetre *Fitxer_MUMs és* una cadena de caràcters que indica el path del fitxer de MUMs a corregir.

**Dades de sortida:**

- Fitxer de MUMs corregit.

#### 4.2.4.4 Programa de generació de SMUMs a partir de MUMs

Programes pel càlcul de SMUMs. Tant el lanza_smums.cc com smums.cc em van ésser proporcionats pel IBB.

**lanza_smums.cc i smum.cc:**

**Descripció general:**

El programa lanza_smums.cc és l'encarregat d'automatitzar el llançament del càlcul de SMUMs per calcular tots els MUMs d'una carpeta. Per poder obtenir els resultats conforme es van generant els MUMs sense haver d'esperar a que acabin tots els càlculs, vaig realitzar unes adaptacions. Passant per paràmetre els valors ID dels dos genomes del mum que volem comparar, només calcula la parella directo/inverso d'aquesta comparació.

El programa fa la crida al smums.cc per fer el càlcul. Per que l'algoritme funcioni correctament, el fitxer de MUMs ha d'estar ordenat de menor a major per la posició inicial del genoma 1.

També vaig modificar els paths dels fitxers factors per que no es solapessin quan es crides el programa un altre cop.

**Procediment de compilació:**

-Cridant al compilador:

g++ lanza_smums.cc -o lanza_smums

gpp smum.cc –o smum

**Procediment d'execució:**

./lanza_smums <Carpeta_Fitxer_MUMs> <ID_genoma1> <ID_genoma2> [<Multiplicador>]

./smum <Nom_Fitxer_MUM> <Multiplicador>

**Dades d'entrada:**

- El paràmetre *Carpeta_Fitxer_MUMs és* una cadena de caracters que indica el path de la carpeta on es troben els MUMs.
- El paràmetre *ID_genoma1* és un numero que identifica el genoma 1.
- El paràmetre *ID_genoma2* és un numero que identifica el genoma 2.
- El paràmetre *Multiplicador* és un valor que indica la tolerància a l'hora de fer nous SMUMs. Si no s'indica, s'utilitza un valor de 2,5.
- El paràmetre *Nom_Fitxer_MUM* és el fitxer MUM del que es calcularan els SMUMs.

**Dades de sortida:**

- Fitxers de SMUMs.
- Fitxer factors per cada parella de MUMs.
- Fitxer factorsMesNoAbsorvits per cada parella de MUMs.

| | **Fitxer exemple de SMUMs** | | |
|---|---|---|---|
| SMUM 1 | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del SMUM |
| SMUM 2 | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del SMUM |
| … | … | … | … |
| SMUM N | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del SMUM |
| | Linia divisoria en blanc, separa SMUMs de MUMs no absorvits | | |
| MUM 1 | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del MUM |
| MUM 2 | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del MUM |
| … | … | … | … |
| MUM M | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del MUM |

| **Fitxer exemple de factors.txt** | | |
|---|---|---|
| Genoma 1 | Genoma 2 | Sumatori de la mida de MUMs i SMUMs dels dos genomes. |
| Genoma 1 | Genoma 3 | Sumatori de la mida de MUMs i SMUMs dels dos genomes. |

## 4.2.4.5 Programa de correcció de SMUMs duplicats

Programa per transformar el fitxer de SMUMs resultant d'un càlcul de SMUMs al format del fitxer de MUMs per realitzar la següent passada de càlcul de SMUMs. Elimina els SMUMs duplicats que sorgeixen del programa smums.cc al utilitzar multiplicadors grans.

**corredupSmums.cc :**

**Descripció general:**

La base d'aquest programa és la mateixa que el programa de correcció de no-MUMs de l'apartat 5.2.4.3. La diferencia principal, és que en aquest cas hem d'executar el programa dues vegades, una per cada columna, ja que ens poden apareixer duplicats pels dos genomes.

Com acabem ordenant el fitxer per la primera columna, i eliminem l'espai separador, de nou té el format de MUM i podem tornar a buscar SMUMs dintre.

Existeix una variant d'aquest programa. Per l'última passada de SMUMs, necessitem que el fitxer ens quedi en format de SMUM, així que no podem cridar aquest programa d'aquesta forma. Per això hem de cridar a l'script separasmums.sh que ens separa els SMUMs dels MUMs no absorbits i crida el CorredupSmums només pels SMUMs. Després del procés torna a ajuntar els fitxers amb el format correcte.

**Procediment de compilació:**

-Cridant al compilador: g++ corredupSmums.cc -o corredupSmums

**Procediment d'execució:**

./corredupSmums <Fitxer_SMUMs>

**Dades d'entrada:**

- El paràmetre *Fitxer_SMUMs é*s una cadena de caràcters que indica el path del fitxer SMUM a transformar.

**Dades de sortida:**

- Fitxer de MUMs.

### 4.2.4.6 Programa de generació del fitxer de similitud entre genomes tots per tots

Programa de concatenació dels fitxers amb la similitud entre dos genomes. El fitxer factors resultant contindrà el grau de similitud entre tots els genomes. Per cada comparació entre genomes el fitxer factors resultant contindrà els identificadors del dos genomes i el sumatori de la mida de tots els seus MUMs i SMUMs. Podem veure el format al apartat 5.2.4.4

**corredupSmums.cc :**

**Descripció general:**

En aquest programa recorrem la llista de fitxers de la carpeta que s'indica per paràmetre. Per cada fitxer de factors de la comparació de dos genomes, guardem els identificadors dels genomes i ordenem les comparacions pel primer genoma a un vector dinàmic. Finalment generem el fitxer factors amb les comparacions entre tots els parells de genomes.

**Procediment de compilació:**

-Cridant al compilador: g++ uneFactors.cc -o uneFactors

**Procediment d'execució:**

./uneFactors <Carpeta_factors/>

Nota important: A la carpeta_factors només pot haver fitxers de factors temporals i el factors.txt

**Dades d'entrada:**

- El paràmetre *Carpeta_factors é*s una cadena de caràcters que indica el path de la carpeta on podem trobar els factor de semblança resultants de cada comparació

**Dades de sortida:**

- Fitxer factors – Podem veure el format al apartat 5.2.4.4 .

### 4.2.4.7 Script d'optimització del fitxer de SMUMs per ser llegit per l'applet Mummy

Es tracta d'un script que transforma el fitxer de SMUMs per poder ser llegit pel Mummy.

**smumsortV3.sh :**

**Descripció general:**

Aquest script busca en la carpeta de smums/ i per cada fitxer de SMUMs:
Etiqueta els SMUMs i MUMs no absorbits
Ordena per la columna de la mida dels SMUMs i retalla el milió més gran.
Després afegeix un camp d'index d'ordenació per la posició del genoma 2.
A continuació afegeix un camp d'índex d'ordenació per la mida nomes pels SMUMs.

**Procediment de compilació:**
No necessita compilació.

**Procediment d'execució:**
Executant la comanda: sh smumsortV3.sh

**Dades d'entrada:**
- No necessita, només han d'haver els SMUMs a la carpeta smums/ .

**Dades de sortida:**
- Fitxers smumsort a la carpeta smumsort/ .

| | Fitxer exemple de smumsort | | | | |
|---|---|---|---|---|---|
| SMUM 1 | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del SMUM | Ordenació per genoma 2 | Ordenació per mida |
| MUM no absorvit 1 | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del SMUM | Ordenació per genoma 2 | |
| … | … | … | … | ... | ... |
| SMUM N | Posició inicial genoma 1 | Posició inicial genoma 2 | Mida del SMUM | Ordenació per genoma 2 | Ordenació per mida |

## 4.3 Com veure els fitxers de comparació de genomes eucariota amb l'applet Mummy

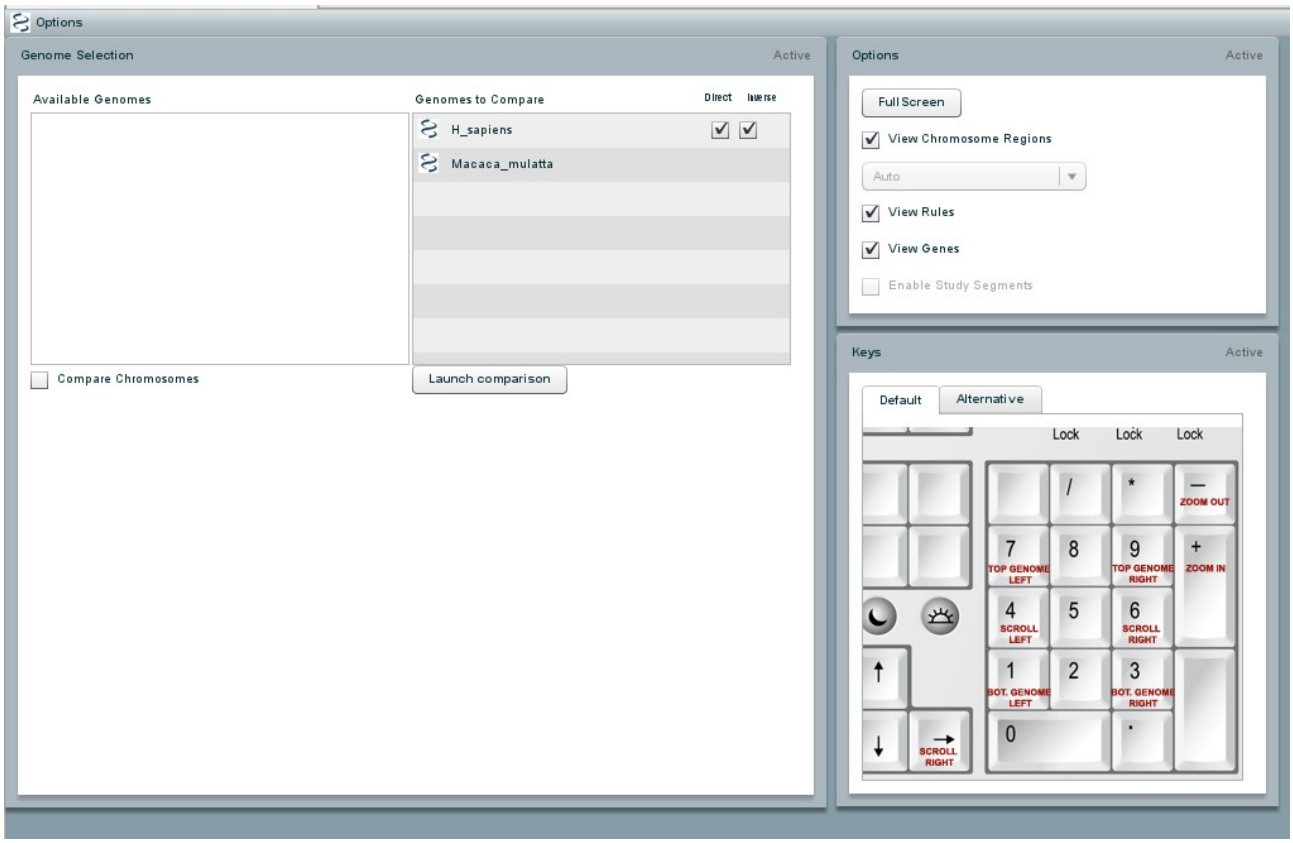
Ara veurem com poder observar els fitxers calculats amb la aplicació web Mummy. Necessitem que els fitxers de SMUMs amb ordenació existeixin a la carpeta smumsort/ corresponent al path que passem al atribut "path". El mateix amb els fitxers de gens, han d'existir a la carpeta mapedgenes corresponent al path.

Si tot és correcte, existeix una aplicació (que comentaré en el apartat de treball futur) que falta ésser adaptada per eucariotes i que generarà un link com el següent:

http://platypus.uab.cat/genomas/Mummy/Mummy.html?path=..%2Fgenomas%2FEukaryota&V1=1&NombreVar1=H_sapiens&V2=2&NombreVar2=Macaca_mulatta&smumsOnly=true

Fixem-nos en l'adreça:
-L'aplicatiu Mummy que estem fent servir és a /var/www/html/genomas/Mummy/
-Indiquem que el path on ha d'anar a buscar els genomes és a ../genomes/Eukaryota/
-La comparació que estem visualitzant és la corresponent a genoma 1 (V1=1) contra el genoma 2 (V2=2). És a dir, estem carregant els gens 1.gen i 2.gen i els smumsort 1_2.directo.smum i 1_2.inverso.smum .

Quan carreguem l'aplicatiu veiem la següent pantalla, [Figura 27]. Si fem click a "launch comparison" anirem a la pantalla de comparació.




[Figura 27] A la imatge podem veure l'aplicatiu Mummy a la pantalla inicial de opcions. Veiem els noms dels genomes a comparar, si volem vista de cromosomes o de genoma les opcions de visualització i instruccions del teclat per poder-nos desplaçar.

# 5. Conclusió

Tot i els problemes trobats durant la realització del treball hem aconseguit complir els objectius dins el termini plantejat. Hem aconseguit l'automatització total del procés de comparació de genomes eucariota en un servidor públic a l'abast de tota la comunitat científica. Des de el moment en que apareix un nou genoma, o s'actualitza un genoma incomplet al repositori mundial de genomes (NCBI) fins que vulguem veure la comparació de aquest amb qualsevol altre genoma disponible.

Una de les parts més importants que vàrem aconseguir va ser fer viable la comparació de genomes d'eucariotes, fent la comparació per a cada cromosoma d'un genoma contra un altre genoma sencer. La adaptació del calcul de MUMs va suposar diversos reptes que vam poder superar:

-Comparem sempre el genoma amb cromosomes més petits, partit en cromosomes. D'aquesta manera tenim suficient memòria RAM per realitzar el màxim de comparacions simultànies.

-Desenvolupament d'un sistema per concatenar els MUMs de cromosomes resultants de la comparació de cada cromosoma. Això implicava eliminar els MUMs inclosos a un altre cromosoma i que per tant no eren MUMs, i posicionar els MUMs definitius al genoma. Un problema afegit va ser que estem treballant amb dades reals no preparades per a operar de forma automàtica sobre elles. Per aquest motiu trobar l'ordre dels cromosomes dins del genoma, i fer-lo de forma automàtica, ha sigut tot un repte.

Realitzar la paral·lelització al calcul de MUMs va ser un dels temes que més em va agradar. Disposava d'un servidor molt potent amb 24 processadors i havia d'aplicar alhora diversos conceptes vists durant la carrera per aprofitar-lo al màxim i generar els càlculs correctes. Havia de repartir entre els processadors tots els càlculs de MUMs que la memòria ens permetés . Això és per que treballant amb eucariotes la nostre limitació és la memòria, pel que vaig haver de explotar tant funcions de C++, com conceptes d'administració GNU/Linux, per desenvolupar un sistema òptim, funcional i sense errades. A trets generals, vam aconseguir una millora aproximada en el temps d'execució de un 600%. Sempre depenent del mida i quantitat dels cromosomes de les especies que comparem, que poden fer variar la millora entre un 400% fins a un 1000%. Per posar un exemple, la comparació del toro o bos taurus amb el peix danio rerio trigava aproximadament 20 hores i actualment amb la paral·lelització em aconseguit un calcul en 3 hores i quart, és a dir, sis vegades més ràpid.
Aquesta paral·lelització ha implicat la correcta modificació de noms de fitxers per evitar solapacions, l'ús de semàfors per la sincronització si diversos processos havien de utilitzar un mateix recurs, l'ús de forks, waits, senyals, consultes al sistema mitjançant la lectura de fitxers en memòria del /proc, etc.

L'adaptació a eucariotes del procés de generació de SMUMs va ser molt llarg i tediós. El treballar amb dades reals tant grans feia molt difícil trobar els errors dels fitxers generats: SMUMs que es repetien, fitxers que no carregaven al aplicatiu web Mummy, execucions que no terminaven mai, etc. Tot plegat, em va portar moltes hores de proves saber quin era el procediment a seguir perquè tot funcionés adequadament.

En la fase de càlcul de mamífers, va resultar bastant fàcil l'automatització del programa encarregat de generar el mapatge de gens. Una petita millora al codi per no tractar les versions alternatives de genomes que no ens interessaven, va ser suficient. En la fase de càlcul de totes les especies animals es va complicar l'automatització. Al sortir de mamífers, van aparèixer nous tipus de cromosomes que calien ésser processats. Van aparèixer noves formes de indicar la posició dels gens. I el que més mals de caps em va donar, al repositori NCBI, a alguns genomes no els fan seguir una nomenclatura o numeració estàndard, pel que vaig haver de modificar el parser del programa per acceptar qualsevol tipus de nom que pogués venir.

Per el programa de descarrega automàtica de genomes vaig desenvolupar un programa en java. Ho vaig trobar el més adequat per la facilitat al crear connexions FTP i gestionar-les. Vaig aconseguir obtenir les dates de modificació dels fitxers dels genomes per actualitzar-los quan fos necessari. Vaig utilitzar les eines de consultes a les bases de dades del NCBI, les e-utils, per aconseguir informació sobre els nous genomes trobats al servidor del NCBI. Aquestes retornen la informació en un XML, on el java dóna moltes facilitats per parsejar aquests tipus de fitxers. Treien alguns problemes per obtindre la informació que necessitava de les Bases de dades del NCBI en cada moment, va ser interessant la possibilitat de treballar amb diverses tecnologies alhora i utilitzar-les per finalment aconseguir l'objectiu plantejat.

Un altre fase on vaig gaudir molt, va ser en la programació del “robot” automàtic. Va ser una fase de unió de tots els programes que havia estat desenvolupant durant tot el treball i era aparentment una cosa senzilla. Em va servir per veure tots el problemes de compatibilitat que poden sorgir en una unió tan gran de programes. Després de fer una serie de adaptacions, el fet veure com tots els “engranatges” giraven a les ordres del robot va ser molt gratificant. La feina estava feta.

Finalment, després de tot el que he aprés, tot el que he gaudit i perquè no, tot el que he patit, voldria expressar el meu desig que el treball desenvolupat sigui d'un pes important als avenços de la biotecnologia i permeti fer nous descobriments gràcies a la comparació de similituds i diferències entre els genomes de les diferents espècies que a donat l'evolució.

## 5.1 Treball futur

A partir del fitxer de similituds de genomes que genera el càlcul de SMUMs, podem visualitzar amb un programa en java disponible al servidor, l'arbre de genomes. Aquest arbre mostra tots els genomes comparats i distanciats segons la relació de similitud que tenen.

Aquest programa ha d'agafar automàticament el fitxer de similituds per generar l'arbre, i quan l'usuari faci un click als genomes, el programa ha de fer una crida a l'aplicació web Mummy per visualitzar la comparació dels genomes en detall.

Existeix una forma d'optimitzar més l'ús de memòria, permetent així la possibilitat de comparar més cromosomes alhora i guanyant en temps d'execució. La dificultat d'aplicar la idea, i el temps limitat del que disposàvem, no ens va permetre realitzar-la. Al moment de fer la comparació de dos genomes (suposem que de 3 Gb), cada cromosoma es compara contra el segon genoma complet. Si caben a memòria 4 comparacions de cromosomes, estem carregant 4 vegades el segon genoma a memòria, ocupant 12 Gb en comptes de 3Gb que necessitem. A més ens estalviariem el temps de treball del parser del segon genoma. Aquesta idea s'ha d'aplicar al MUMOL i adaptar-lo per carregar el segon genoma al inici de la comparació, i no alliberar-ho fins al final de la mateixa.

# 6. Referències


[1] **Web oficial NCBI** , (National center for biotechnology information) Repositori mundial de biotecnologia.
http://www.ncbi.nlm.nih.gov/

[2] **FTP NCBI** , Conté els genomes actualitzats i tota la informació que necessitem per realitzar tot el procés de comparació.
ftp://ftp.ncbi.nih.gov/

[3] **Entrez Programming Utilities Help** , ajuda on-line per l'us de les consultes amb e-utils
http://www.ncbi.nlm.nih.gov/books/NBK25501/

[4] http://platypus.uab.cat/ , Web server for the all-known-genomes comparison by web. Server supported by the Institute of Biotechnology and Biomedicine of the Autonomous University of Barcelona (IBB-UAB).

[5] **Efficient Space and Time multicomparision of Genomes**, Mario Huerta, Xavier Messeguer, Technical report LSI-02-64-R. Llenguatges i Sistemes Informatics, Universitat Politècnica de Catalunya (2002).

[6] **Identification of patterns in biological sequences at the ALGGEN server: PROMO and MALGEN**, Domènec Farré, Mario Huerta, Romà Roset, José E. Adsuara, Llorenç Roselló, M. Mar Albà, and Xavier Messeguer, Nucleic Acids Research. 2003 31: 3651-3653 (2003).

[7] **M-GCAT: Interactively and efficiently constructing large-scale multiple genome comparison frameworks in closely related species.** T. Treangen and X. Messeguer. BMC Bioinformatics 2006, 7:433.(2006)

[8] **Suffix Tree Construction with slide nodes,** Mario Huerta. technical report LSI-02-63-R Dep. Llenguatge i Sistemes Informàtics, Universitat Politècnica de Catalunya (2002).

[9] http://ibb.uab.cat/ibb/ , Web de l'Institut de Biotecnologia i de Biomedicina.

[10] http://cic.puj.edu.co/wiki/doku.php?id=materias:laboratorio_de_lenguajes_ii:lableng2:regexp , Web amb informació de expressions regulars

[11] http://redes-privadas-virtuales.blogspot.com/2009/07/liberar-la-memoria-ram-cacheada-en.html , Web amb informació per alliberar la memòria RAM que no s'utilitza.

[12] http://geekalt42.net/el-porque-no-usar-un-task-killer-en-android-optimizando-la-ram-3813 , Web am informació referent a la optimització de memòria RAM.

[13] http://www.chuidiang.com/clinux/ipcs/semaforo.php , http://www.infor.uva.es/~isaac/SO/practicas/guiones/semaforos-memoria.html , informació de programació amb semàfors.

[14] http://linux.die.net/man/2/wait ,
http://www.csl.mtu.edu/cs4411/www/NOTES/process/fork/wait.html ,
http://osr507doc.sco.com/en/man/html.C/wait.C.html , manpage i informació de les funcions fork i wait.

[15] http://www.eclipse.org/ , pàgina oficial del projecte eclipse.

[16] http://www.enterprisedt.com/products/edtftpj/ , llibreries per la connexió FTP via java.

[17] http://www.cplusplus.com/ , Recurs imprescindible per la programació en C++.

[18] http://download.oracle.com/javase/6/docs/api/ , Recurs imprescindible per la programació en Java.

# 7. Resum

## 7.1 Català

La cerca de similituds als codis genètics de dos espècies, ens permet obtenir molta informació de la evolució dels seus genomes. Aquesta informació afavoreix el descobriment gens que es conserven amb la mateixa funcionalitat a diferents espècies. També té importants aplicacions mèdiques i ens permet entendre els processos evolutius que han portat a la diversitat d'espècies de l'actualitat.

El present treball té l'objectiu d'automatitzar una serie de processos d'un servidor d'aplicacions web: http://platypus.uab.cat, que realitzin de forma òptima i eficient, la comparació dels genomes eucariotes, tots amb tots, conforme aquests genomes siguin seqüenciats. Així aquestes comparacions entre genomes de organismes superiors podran ser consultades via web.

## 7.2 Español

La búsqueda de similitudes en los códigos genéticos de dos especies, nos permite obtener mucha información de la evolución de sus genomas. Esta información favorece el descubrimiento de genes que se conservan con las mismas funcionalidades en diferentes especies. También tiene importantes aplicaciones médicas y nos permite entender los procesos evolutivos que han llevado a la diversidad de especies en la actualidad.

El presente trabajo tiene como objetivo automatizar una serie de procesos de un servidor de aplicaciones web: http://platypus.uab.cat/ , para que realicen de forma óptima y eficiente, la comparación de los genomas eucariotas, todos con todos, conforme esos genomas vayan siendo secuenciados. Así esas comparaciones entre genomas de organismos superiores podrán ser consultadas vía web.

## 7.3 English

The search for similarities in the genetic codes of two species lets us obtain a lot of evolution information of their genomes. This information favors the discovery of genes that are kept with the same functionalities in different species. It also has important medical applications and lets us understand the evolutionary processes that have led to the diversity of species we have nowadays.

This project has as its main objective to automate a series of processes of an applications web server: http://platypus.uab.cat/ to make, in an ideal and efficient way, a comparison between all of the eukaryotic genomes, as they get arranged and organized. So, those comparisons between genomes of superior organisms can be looked up through the web.