Software verslag databases 2

P. Dopheide
Software verslag databases 2
Bio-informatica (BFV-4)
Groningen, 20-04-2012
Leraar: S. Warris

Hanzehogeschool Groningen
Institute for Life Science and Technology
Year 2011-2012

Inhoudsopgave

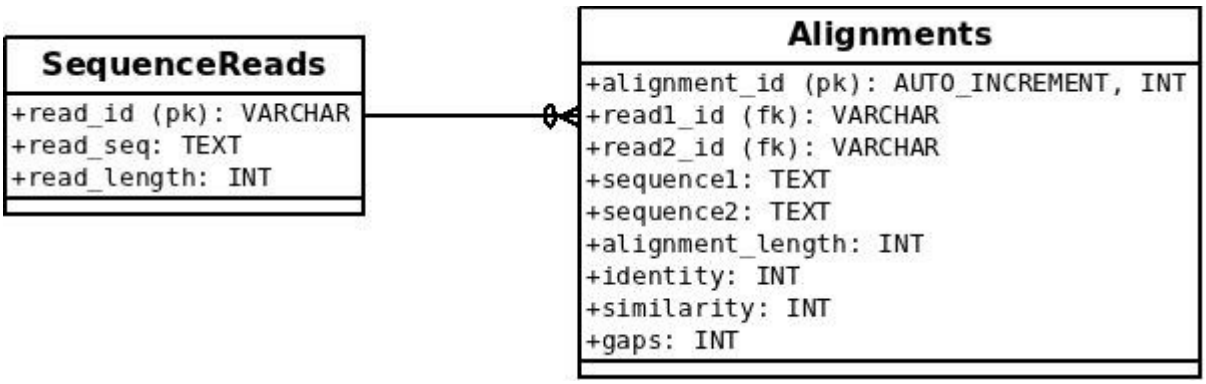
Inleiding

Dit project draait om het opslaan en verwerken van sequence read alignment informatie. Hiervoor word een database gemaakt voor het opslaan van de sequence reads en de alignments tussen twee reads. De database moet gevuld kunnen worden vanuit een script en er moeten relevante vragen gesteld kunnen worden met een script.

Met behulp van een script moet de gebruiker in staat zijn de volgende vragen te stellen:
– Gegeven een read, met welke andere read is er de meeste overlap?
– Geef alle reads die een exacte match hebben.
– Geef alle reads waarbij er 1 SNP is.

Database ontwerp

Voor dit project is de database zo simpel mogelijk gehouden en bestaat daarom, zoals te zien in figuur 1, uit maar twee tabellen. De database en stored procedures zijn gemaakt in MySQL versie 5.1.61-0+squeeze1.



Figuur 1. Ontwerp van de database.

SequenceRead tabel
De eerste tabel is de SequenceReads tabel. Deze tabel is ontworpen om de korte sequence reads en hun eigenschappen op te slaan. Van de reads word er een ID opgeslagen in een varchar veld om ze te herkennen en terug te vinden. Er is voor een varchar veld gekozen omdat de ID uit het fasta bestand gehaald word en deze ID's vaak een combinatie van letters, cijfers en speciale tekens (zoals “|”) bevat. Naast de ID word ook de nucleotide sequentie zelf opgeslagen. Deze kan bijvoorbeeld gebruikt worden om het CG% percentage te herleiden mocht dit in een onderzoek nodig zijn. De sequentie word bewaard in een text veld, dit omdat ook “korte” sequenties voor sommige nog 900 base of meer lang kunnen zijn. Het laatste veld in de SequenceReads tabel bevat de lengte van de read, wat opgeslagen word als een int.

Alignments tabel
De tweede tabel waar de database uit bestaat is de Alignments tabel. Deze tabel is beschrijft een alignment tussen twee sequenties. Een alignment heeft een ID nodig als identifier, in dit geval is dat een int die oploopt per alignment die aan de database wordt toegevoegd. Een alignment bestaat uit twee sequenties, daarom worden de ID's van beide sequenties opgeslagen als foreign keys zodat ze uit de SequenceReads tabel terug te halen zijn. Ook worden de gealignde sequenties opgeslagen, deze sequenties kunnen na het alignen gaps bevatten (weergegeven met een “-”) daarom is er ook hier voor een text veld gekozen. Naast de twee sequenties die gealigned zijn heeft een alignment nog andere eigenschappen. Van deze eigenschappen worden lengte van de alignment, de identity en similarity tussen de twee sequenties en de hoeveelheid gaps in de alignment opgeslagen. Deze eigenschappen worden allemaal beschreven met een getal, waardoor voor alle velden gekozen is voor een int.

Relatie
De relatie tussen beide tabellen is 1 op 0, 1 of meer. Een read kan niet (voldoende) alignen tegen een andere read, waardoor er geen alignment is, een alignment hebben met 1 andere read of meerdere alignments van voldoende kwaliteit tegen meerdere reads.

Stored procedures

Naast de sql code voor het maken van de database zijn er ook stored procedures geschreven. Deze stored procedures worden door de python code gebruikt bij het plaatsen van data in de database en bij het bevragen van de database.

De stored procedures zijn verdeeld over twee bestanden, fill_db_storedproc.sql en ask_db_storedproc.sql. In fill_db_storedproc.sql staan de procedures read2db en alignment2db voor het plaatsen van reads en alignments in de database respectievelijk. In ask_db_storedproc.sql staan de procedures GetMostOverlap, GetExactMatch en GetOneSNP. Deze procedures worden door de code gebruikt voor het bevragen van de database.

Ontwerp scripts

Voor dit project zijn er meerdere scripts geschreven voor het uitvoeren van verschilende taken, deze staan hieronder beschreven. De scripts zijn geschreven in Python versie 2.6.6.

db2_main.py
Een van de twee “hoofd scripts”. Dit script roept de andere scripts aan die nodig zijn voor het plaatsen van de reads in de database, het alignen van de reads en het plaatsen van de alignments in de database. Het script bevat het pad naar een multifasta bestand wat het doorgeeft aan het fasta_parser script (zie hieronder) waarvan het de geparstee inhoud van het bestand weer terug krijgt. De inhoud wordt doorgegeven aan de read2db functie. Deze functie loopt alle sequenties bij langs en plaatst ze in de database met behulp van het mysql_dao script (zie hieronder). Nadat alle reads in de database zijn geplaatst wordt de functie ready_reads aangeroepen waaraan de inhoud van het fasta bestand wordt mee gegeven. Deze functie loopt alle sequenties langs met een for loop. In de for loop zit een tweede for loop die ook alle sequenties afgaat maar die begint bij de sequentie na de huidige sequentie in de eerste for loop. Op deze manier worden er steeds twee sequenties genomen, een per loop, die doorgegeven worden aan het aligner script (zie hieronder), waardoor beide sequenties tegen elkaar gealigned worden. Wanneer de alignment klaar is word het alignment_parser script (zie hieronder) aangeroepen die het alignment bestand parsed en de alignment in de database zet.

fasta_parser.py
Het fasta_parser script verwacht bij aanroepen een te openen (multi)fasta bestand. Met de parse_fasta functie zal geprobeerd worden het bestand te openen, waarna het bestand geparsed word met de SeqIO.parse() functie uit de Bio module van biopython. De inhoud van het bestand word in een lijst geplaatst en geretouneerd.
Het script bevat ook een main functie die aangeroepen word als het script op zichzelf gedraaid word. Deze functie is gebruikt voor testen en kan aangepast/uitgebreid worden om een los staande fasta parser te maken.

aligner.py
Het aligner script verwacht twee sequenties. Deze sequenties worden opgeslagen in twee tijdelijke bestanden. Na het opslaan van de sequenties wordt er een command line regel voorbereid voor het draaien van het alignment programma needle van Emboss (http://emboss.sourceforge.net/). Het voorbereiden van de command line regel wordt gedaan met de NeedleCommandline() functie afkomstig van biopython. De alignment word gestart met behulp van de subprocess modulle, waarna er gewacht wordt totdat de alignment klaar is. De resulterende alignment uit needle word opgeslagen in een tijdelijk bestand en terug gegeven.

alignment_parser.py
Het alignment_parser script leest een alignment bestand in, parsed de inhoud en schrijft de alignment naar de database met behulp van het mysql_dao script (zie hieronder). Eerst word het alignment bestand ingelezen, waarna het doorzocht word met een drietal regular expressies, of regexen. Deze regexen zoeken naar de identity, similarity en hoeveelheid gaps. De resultaten hiervan worden in lijsten opgeslagen, die daarna samen worden gevoegd in een lijst zodat de waarden makkelijk doorgegeven kunnen worden. Daarna word het alignment bestand geparsed met behulp van AlignIO.read van biopython, deze functie parsed tevens ook de alignment. De geparste alignment word doorgegeven samen met de lijst met de regex data aan de functie alignment2db die de alignment en bijbehorende characteristieken in de database plaatst.

ask_db.py
Dit script bevat verschillende functies voor het bevragen van de database. Bij het starten van dit script word de main functie aangroepen. De main functie vraagt aan de gebruiker welke van de volgende drie vragen de gebruiker wil stellen. Vragen die gesteld kunnen worden zijn:
– Gegeven een read, met welke andere read is er de meeste overlap?
– Geef alle reads die een exacte match hebben.
– Geef alle reads waarbij er 1 SNP is.
Afhankelijk van de invoer van de gebruiker wordt een van de volgende functies aangeroepen, ask_read_id, overlap, exact of snp. Deze functies roepen een functie in het mysql_dao script aan die de database bevraagd met de gewenste vraag en de resultaten terug geeft. De functie geeft dan de resultaten op de command line weer of, als de query geen resultaten levert, wordt hier een melding van gegeven aan de gebruiker.

mysql_dao.py
Het mysql_dao script bevat meerdere functies voor het maken van een connectie met de database en het plaatsen en opvragen van data uit de database. Het heeft twee private functies, een voor het maken van een connectie met de database en een voor het geven van de cursor. Voor het plaatsen en opvragen van data naar en van de database zijn de functies get_row, get_rows en execute. De get_row functie bevraagd de database met een meegegeven query en geeft de eerste regel van het resultaat terug. De get_rows functie heeft de zelfde functionaliteit als de get_row functie behalve dat het niet een enkele rij maar meerdere rijen terug geeft. De execute functie voert een meegegeven query, zoals het plaatsen van data, uit. Daarnaast heeft mysql_dao ook de functies store_read, store_alignment, get_most_overlap, get_exact_match en get_one_snp. Met de store functies word een read met bijbehorende gegevens, de ID en lengte, opgeslagen in de database of een alignment met bijbehorende gegevens opgeslagen. Met de get functies word de database bevraagd op een van de vragen die beschreven staan bij ask_db.py en worden de resultaten geretouneerd. Alle queries die uitgevoerd worden maken gebruik van een van de hierboven beschreven stored procedures.

Bijlage

Database code
Stored procedures code
Python scripts