Snakemake ve Conda Kullanılarak Otomatik Veri Analizi

Dr. Emrah Akkoyun, Dr. Ogün Adebali

TÜBİTAK-ULAKBİM, Sabancı Üniversitesi Mühendislik ve Doğa Bilimleri Fakültesi

EuroCC Türkiye Seminerleri
16 Eylül 2021, Zoom Toplantı

TEAM

Emrah Akkoyun

Nurdan Kuru

Aylin Bircan

Onur Dereli

SUPERVISORS:

OGÜN ADEBALİ
Principal INVESTIGATOR

ÖZNUR TAŞTAN
Principal INVESTIGATOR

Bilgi

- **6 uygulama**
- **Her bir uygulama klasörü içerisinde**
  - Readme dosyası (komutlar, çıktıları)
  - Uygulama için gerekli tüm dosyalar (iş dosyası, girdi, çıktı, snakefile, log vb)
- **Kaynak**
  - https://docs.truba.gov.tr/how-to-guides/Snakemake/index.html
  - Tüm uygulamalar
  - Kapsamlı rehber dosyası (uygulama + bilgi)
- **Uygulama yaparken**
  - Slurm işlerinde hesap bilgisi (account) değiştirilmeli
  - Snakemake ile iş gönderirken
    - Çalışma dizini (workdir)
    - Hesap bilgisi

İÇERİK

Giriş

- **Amaç**
  - Ortak dil oluşturma
  - Basit bir iş koşturma
- **Temel bileşenler**
  - Kullanıcı arayüzü (levrek1)
  - İş yükü yöneticisi (slurm)
  - Hesaplama ucu (barbun, sardalya)
  - İş (hesaplama)
- **İşin özellikleri**
  - Kaynak gereksinimi (işlemci, bellek, vb)
  - Çalışma ortamının hazırlanması (kurulum, çevre değişkenleri, vb)
  - İşi koşturma ve yönetme (gönderme, çıktı üretme, loglama)

Uygulama -1: YBH kümesinde örnek iş koşturma (exp1)

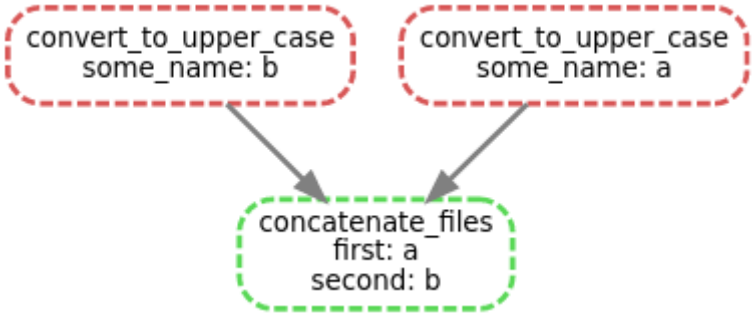
Uygulama -1: YBH kümesinde örnek iş koşturma

Uyarılar

- Kullanıcı arayüzü üzerinde iş koşturmamak
- İhtiyaca yönelik kaynak talebinde bulunmak
  - Node ve çekirdek sayısı

Karmaşık süreçli veri analizi, zorluklar

- **Amaç:**
  - Tek bir iş dosyası, birden fazla hesaplama
    - Örnek
    - Zorluklar
  - Paket (yazılım, kütüphane, araç) kurulumu
- **Bilgi**
  - Hesaplama (görev)
    - Girdi, çıktı, çalıştırılabilir kod, log, araç/yazılım
    - Kaynak gereksinimi
    - Ortam değişkenleri
  - İş akışı
    - Ardışık hesaplama
  - Ölçeklenebilirlik & Tekrar üretebilirlik & Mükerrer hesaplama



Source: SciLifeLab, National Bioinformatics Infrastructure Sweden (NBIS)

Uygulama -2: Ardışık hesaplamanın yapıldığı örnek iş koşturma (exp2)

Uygulama -3: Ardışık hesaplamanın ölçeklenebilirlik (exp3)

Karmaşık süreçli veri analizi, zorluklar

- **Süper kullanıcı (root) olmadan kurulum yapma**
  - Kaynak kod üzerinden derlemek
  - Her bir pakete özgü direktifleri takip etmek
  - Kütüphane/yazılım bağımlılığı
  - Dikkatli işlemci mimarisi/derleyici seçimi
  - Taşınabilirlik, tekrar üretebilirlik sorunları
  - Normal bir kullanım için zahmetli ve zaman alıcı
- **Takip edilmesi zor iş akışı**
  - Girdi ve çıktı dosyalarını takip etmek
  - Araya yeni bir hesaplama eklemek/çıkarmak
- **Mükerrer hesaplama**
- **Farklı iş karakteristiği, aynı kaynak tahsisi**
  - Örn: işlemci yoğun, bellek yoğun hesaplamalar
- **Esneklik sorunları**
  - Paket yükseltme
- **Ölçeklenebilirlik sorunları**
  - Her bir protein için yeni iş dosyası hazırlamak
  - Tüm çıktı/log dosyalarının saklanması ve yönetilmesinde ki zorluklar

İş akışı yöneticisi ve avantajlar

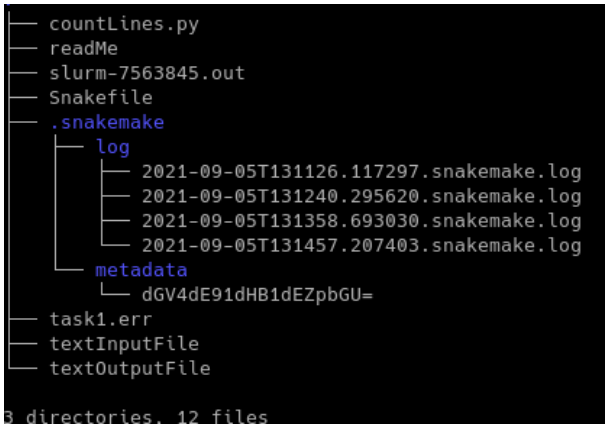
Amaç

- Basit bir iş akışı oluşturma ve küme üzerinde koşturma
- Paket yöneticisi (conda) kurulum ve entegrasyon
- Ölçeklenebilirlik

Basit iş akışı (Snakefile)

- Okunabilir yapılandırma (json)

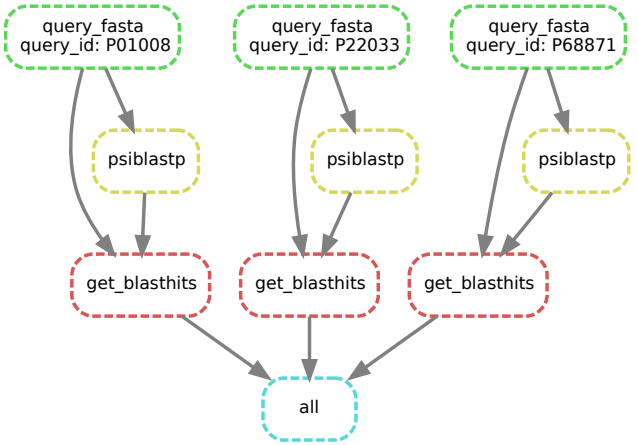
```
1  rule task1:
2          input:
3                  text_file = "textInputFile"
4          output:
5                  text_output = "textOutputFile"
6          log:
7                  "task1.err"
8          shell:
9                  "python countLines.py {input.text_file} {output.text_output} 2 > {log}"
```



İş akışı yöneticisi ve avantajlar

Çoklu hesap

- 3 protein ve her biri için 1 hesaplama
- Örnek iş akışı diagramı
- Toplam 10 hesaplama
  - Slurm işi
  - Kaynak tahsisi
  - Paket seçimi (conda)
  - Loglama & Performans
- Mükerrer hesaplamanın önüne geçme
- Json tabanlı config dosyasını ayırma
- Çıktıların (log, output, benchmark) organize tutulması




```
├── config.yml
├── envs
│   ├── blastp.yml
│   └── python.yml
├── logs
│   ├── P01008_get_blasthits.err
│   ├── P01008_psiblastp.err
│   ├── P01008_query_fasta.err
│   ├── P22033_get_blasthits.err
│   ├── P22033_psiblastp.err
│   ├── P22033_query_fasta.err
│   ├── P68871_get_blasthits.err
│   ├── P68871_psiblastp.err
│   └── P68871_query_fasta.err
├── output
│   ├── P01008
│   │   ├── P01008_blasthits.fasta
│   │   ├── P01008_blasthits.out
│   │   └── P01008.fasta
│   ├── P22033
│   │   ├── P22033_blasthits.fasta
│   │   ├── P22033_blasthits.out
│   │   └── P22033.fasta
│   └── P68871
│       ├── P68871_blasthits.fasta
│       ├── P68871_blasthits.out
│       └── P68871.fasta
├── readMe
├── scripts
│   ├── make_query_fasta.py
│   └── parse_blastp.py
├── slurm-7565694.out
├── slurm-7565695.out
├── slurm-7565696.out
├── slurm-7565728.out
├── slurm-7565739.out
├── slurm-7565741.out
├── slurm-7565752.out
├── slurm-7565780.out
├── slurm-7565863.out
├── slurm-7565868.out
├── Snakefile
└── workflow.svg
```

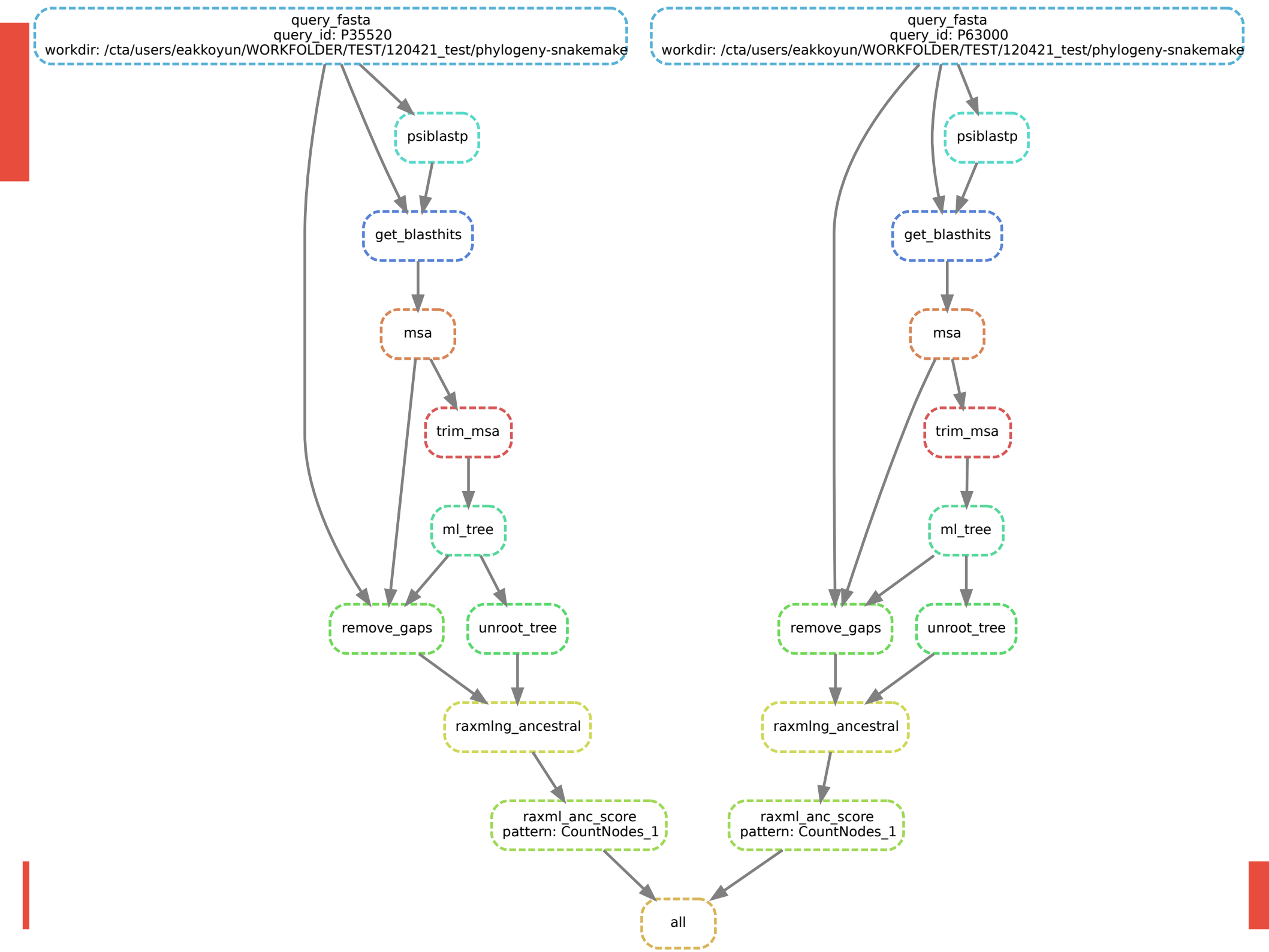
Uygulama -4: Tek hesaplı basit iş akışı örneği (exp4)

Uygulama -5: Çok hesaplı basit iş akışı örneği (exp5)

Uygulama -6: Snakemake ile gerçek uygulama - Phylas (exp6)

İleri düzey yapılandırma (snakemake & conda)

- **Amaç**
  - Karmaşık süreçli veri analizi gerçek örnek - Phylas
- **Avantajlar**
  - Taşınabilirlik (slurm profile, truba & sabancı)
  - Ölçeklenebilirlik (config dosyası; query_ids, pattern)
  - Tekrar üretebilirlik (conda envs)
  - Dağıtabilirlik (Dosya-dizin organizasyon)
  - Önbellekleme (rule dosyaları, parametre)
  - Loglama & Performans (rule dosyaları; logs & benchmarks)
  - İzleme, raporlama (Panoptes, wms-monitor)
  - Python kodu çalıştırabilme (Snakefile)
  - Singularity (container)
  - Entegrasyon (S3 API)


query_fasta
query_id: P35520
workdir: /cta/users/eakkoyun/WORKFOLDER/TEST/120421_test/phylogeny-snakemake
query_fasta
query_id: P63000
workdir: /cta/users/eakkoyun/WORKFOLDER/TEST/120421_test/phylogeny-snakemake
psiblastp
psiblastp
get_blasthits
get_blasthits
msa
msa
trim_msa
trim_msa
ml_tree
ml_tree
remove_gaps
unroot_tree
remove_gaps
unroot_tree
raxmlng_ancestral
raxmlng_ancestral
raxml_anc_score
pattern: CountNodes_1
raxml_anc_score
pattern: CountNodes_1
all

SONUÇLAR

- **Karmaşık süreçli veri analizlerinde, bir araç kullanmadan (snakemake, conda) hesaplama yapmak mümkün, ancak hiç pratik değil.**
- **Snakemake & Conda, sadece iş akışı gerektiren hesaplamalar da değil, fazla sayıda iş/kaynak gerektiren tüm hesaplamalarda kullanılabilir.**
- **Hazırlık, öğrenme zaman alıcı bir süreç, uzun vadede sağladığı pek çok avantajla işleri çok kolaylaştırıyor.**
- **Araştırma döngüsünde (iş akışında değişiklik, parametre havuzu, yazılım güncelleme, yeni girdi dosyaları) büyük avantaj sağlar.**

- TRUBA
  - Bayram öncesi, 144 hesaplama ucu, 2x28 çekirdek
  - Docs truba
  - Destek
- SABANCI, ADEBALİ LAB
- TÜBİTAK-ULAKBİM

TEŞEKKÜRLER

Anket için:

shorturl.at/ckA57

Soru ve Görüşleriniz için:

emrah.akkoyun@metu.edu.tr