



Grid aprēķinu vide

teorija • metodes • aprēķini



GRINDER

AUTOMATIZĒTA GRID DARBU PĀRVALDĪBAS SISTĒMA

Dokumenta faila vārds:	LG-nodevums-grinder.doc
Aktivitāte:	2.aktivitāte – Starpprogrammatūras rīku praktiskā realizācija
Projekta numurs:	VPD1/ERAF/CFLA/05/APK/2.5.1./000055/027
Organizācija:	Latvijas Universitātes aģentūra "Latvijas Universitātes Matemātikas un informātikas institūts"
Autori:	Jānis Džeriņš, Kārlis Podiņš, Baiba Kaškina

Anotācija: Grinder ir viegli pielāgojama un paplašināma Grid darbu pārvaldības sistēma, kas nodrošina automātisku Grid darbu palaišanu, kontroli un rezultātu saglabāšanu, kā arī darbu pirmsapstrādi un pēcapstrādi.

Saturs

IEVADS	3
1. GRINDER UZBŪVE UN PAPILDINĀŠANA.....	4
2. UZDEVUMA DZĪVESCĪKLS	4
3. REALIZĒTIE UZDEVUMI.....	5
3.1. STANDARTA UZDEVUMS.....	5
3.2. ANSYS	5
3.2.1. Ievada formāts	5
3.2.2. Izvada formāts.....	6
3.2.3. Uzdevuma pārrēķināšana	6
3.3. SEMTI-KAMOLS	6
3.3.1. Uzdevuma sadalīšana	6
3.3.2. Darbu izpilde	7
3.3.3. Rezultātu apstrāde	8
3.4. GRINDER IZMANTOŠANA	8
3.4.1. Opciju un parametru apraksts	10
4. SECINĀJUMI.....	13

IEVADS

Viena no jomām, kur ir izdevīgi izmantot grid skaitļošanu, ir uzdevumi, kuru atrisināšanai ir vajadzīgs liels procesora laiks. Ja šādus uzdevumus var sadalīt mazākos neatkarīgos apakšuzdevumos un no apakšuzdevumu rezultātiem sintezēt pamatuzdevuma rezultātu, tad ir izdevīgi apakšuzdevumus rēķināt gridā, tādējādi saīsinot galarezultāta gaidīšanas laiku, jo apakšuzdevumus var risināt paralēli.

Uzdevumu sadalīšana apakšuzdevumos un galarezultāta sintēze ir pilnīgi atkarīgi no uzdevuma, bet apakšuzdevumu risināšana grid tīklā – uzdevumu palaišanu, kontroli un rezultātu saglabāšana dažādiem uzdevumiem ir ļoti līdzīga.

Projektā izmantotās grid tehnoloģijas piedāvā funkcionāli pilnvērtīgu servisu klāstu, taču tām ir divi būtiski trūkumi:

- nepietiekami automatizēšanas līdzekļi liela skaita līdzīgu grid darbu apstrādei,
- nav augsta līmeņa rīku, kuru lietotāja saskarne atbilstu mūsdienu lietotāja prasībām.

Ir zināmi vairāki risinājumi, kas piedāvā ērtāku lietotāja saskarni (GridCom, Migrating Desktop). Taču tieši līdzīgu darbu automatizēšanas problēma ir kritiska, lai Grid skaitļošanu varētu efektīvi izmantot uzdevumos, kas prasa lielu skaitu grid darbu (SemTi-Kamols, ANSYS).

Grinder nodrošina automatizētu grid darbu palaišanu un rezultātu saglabāšanu, kas, kopā ar grid darbu pirmsapstrādi un pēcapstrādi, sniedz ievērojamus uzlabojumus, ja grid vidē jārisina liels skaits līdzīgu uzdevumu.

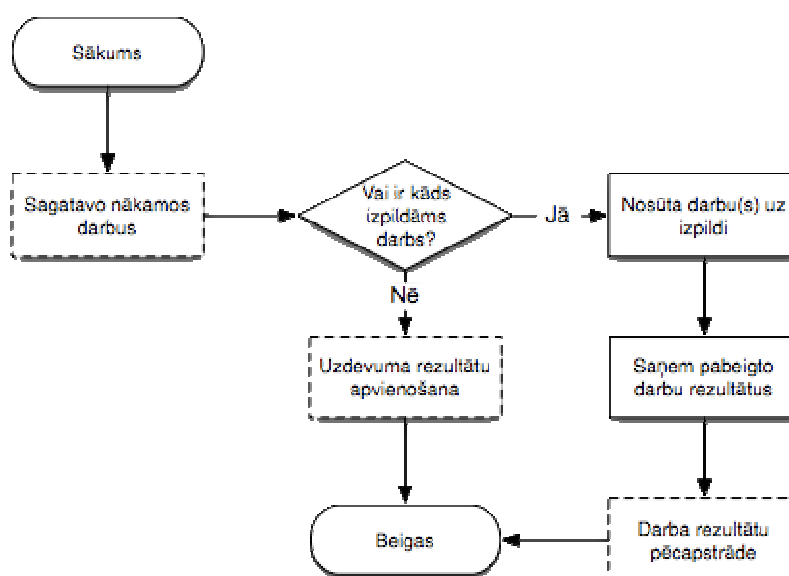
1. GRINDER UZBŪVE UN PAPILDINĀŠANA.

Grinder ir objektorientēta sistēma, kas rakstīta valodā Ruby.

Viens no Grinder pamatjēdzieniem ir uzdevums jeb uzdevuma tips, kam atbilst klase Task. Šajā klasē realizēta grid darbu dzīvescikla funkcionlitāte: darbu nosūtīšana izpildei un rezultātu saņemšana. Darbu sagatavošana (uzdevuma sadalīšana darbos) un pēcapstrāde katram uzdevumam ir atšķirīgas, un tiek realizētas apakšklasēs.

2. UZDEVUMA DZĪVESCIKLS

Grinder vienas izpildes process redzams 1. attēlā:



1. attēls: Grinder izpildes process

Ar raustītām līnijām iezīmētas darbības, kas uzdevumiem ir specifiskas. Visas pārējās darbības visiem uzdevumiem ir kopīgas.

3. REALIZĒTIE UZDEVUMI

Dokumenta tapšanas brīdī Grinder pielāgots kompozītmateriālu konstrukciju simulācijai (izmantojot ANSYS programmatūru) un latviešu valodas tekstu sintaktiskajai analīzei SemTi-Kamols projekta ietvaros.

3.1. STANDARTA UZDEVUMS

Standard uzdevumi ļauj izmantot Grinder piedāvāto funkcionalitāti darbu izpildei un rezultātu saņemšanai, bet ar vienu nosacījumu – uzdevuma dalīšana darbos jāveic lietotājam pašam. Katram uzdevumam *ieejas mapē* tiek izveidota datne ar nosaukumu *inN.zip*, kur N ir attiecīgā darba kārtas numurs. Uzdevuma izpildes laikā šī datne tiek atarhivēta *darba mapē*. Šī darba izpildei iespējami divi scenāriji:

1. Arhīvā ir *darba apraksta datne* (datne ar paplašinājumu *.jdl*) – šī datne tiek izmantota uzdevuma nosūtīšanai izpildei.
2. Arhīvā nav *darba apraksta datnes* – šajā gadījumā arhīvā jābūt divām datnēm:
 - *run.sh*: scenārija datne, kas tiek izpildīta uz izpildes mezgla;
 - *in.zip*: ieejas datu datne, kuras saturs tiek atarhivēts uz izpildes mezgla un pieejams *run.sh* scenārijā.

Tiek sagaidīts, ka darba rezultāti uz izpildes mezgla tiks saglabāti datnē *out.zip* – šī datne arī tiek saņemta kā darba rezultāts un ielikta *rezultātu mapē* ar nosaukumu *outN.zip*, kur N ir tas pats, kas bija darba ieejas datnei.

Šis mehānisms dod iespēju izmantot Grinder funkcionalitāti jebkāda uzdevuma izpildei. *Darba apraksta datni* iespējams izveidot vienreiz un vēlāk izmantot visiem uzdevumiem. Atliek vienīgi sadalīt uzdevuma ieejas datus daļās. Šim mērķim iespējams arī uzrakstīt atsevišķu lietotni jeb scenārija datni – nav obligāti to iekļaut Grinder sistēmā.

3.2. ANSYS

Izmantojot Grinder, tiek rēķināti divu veidu kompozītmateriālu konstrukciju simulācijas uzdevumi – gan ANSYS uzdevumi, gan ANSYS uzdevumi, kas izmanto arī LS_DYNA bibliotēku.

Lielu ANSYS uzdevumu sadalīšanā apakšuzdevumos, kā arī apakšuzdevumu rēķināšana un rezultāta sintēze ir realizēta Grinder sistēmā.

Lai izmantotu Grinder, lietotājam ir vajadzīgs nosūtīt uzdevumus un saņemt rezultātus, pārējās operācijas notiek automātiski. Kā lietotāja saskarni datņu apmaiņai zinātniekam, kas izmanto ANSYS, iespējams izmantot sFTP klientu, tādējādi var izvēlēties sev ērtāko pārlūku, tai skaitā tādu, kam ir mūsdienīga lietotāja saskarne.

3.2.1. Ievada formāts

Ieejas datne ir arhivēta mape, kurā ir 3 datnes – *loop.txt*, *IN.txt* un *USERSAMP.txt* (LS_DYNA uzdevumiem vēl papildus arī *aste.txt*, kas tiek izmantota, lai labotu ANSYS izstrādātāju kļūdu), kas vajadzīgas ANSYS parametriskam uzdevumam. *IN.txt* datnē lietotājam ir iespējams norādīt arī uzdevuma vārdu un parametru, cik smalki lielais uzdevums tiks sadalīts apakšuzdevumos, ierakstot attiecīgi rindas:

```
!*DIVISION=<parametru vektoru skaits katrā apakšuzdevumā>
```

!*TASKNAME=<uzdevuma nosaukums>

3.2.2. Izvada formāts

Rezultātu datnes tiek saglabātas mapē, kuras vārds ir lietotāja definētais uzdevuma nosaukums. Rezultātu datnes ir sanumurētas atbilstoši apakšuzdevumu secībai ievada datnē. Katra apakšuzdevuma rezultāti tiek saglabāti kā atsevišķs .zip arhīvs, .pdrs faili tiek apvienoti vienā failā tālākai kopīgai uzdevuma rezultātu apstrādei un visi ANSYS radītie .jpg attēli tiek apkopoti vienā .zip arhīvā.

3.2.3. Uzdevuma pārrēķināšana

Sagatavotās apakšuzdevumu datnes tiek noglabātas arī atbilstošā rezultātu mapē, apakšmapē *chunks*. Ja izpildē radusies kļūda, tad atbilstošo uzdevumu var pārkopēt uz rēķināmo uzdevumu *darba mapī*, un šis uzdevums tiks izpildīts vēlreiz.

Mazi rezultātu arhīvu izmēri var liecināt par kļūdu izpildē, tāpēc atbilstošie uzdevumi tiek noglabāti arī *rezultātu mapes* apakšmapē *suspicious_chunks*.

3.3. SEMTI-KAMOLS

Šī tipa uzdevumi paredzēti lielu teksta korpusu apstrādei, izmantojot SemTi-Kamols¹ projektā izstrādāto teikumu analīzes un vizualizācijas sistēmu. Ieejas datnes atrodamas vienā mapē (iespējams ar apakšmapēm). Šis uzdevuma tips realizēts klasē *Kamols*, un tas realizē automātisku ieejas datņu sadalīšanu grid uzdevumos, to apstrādi, rezultātu saņemšanu, un pēcapstrādi.

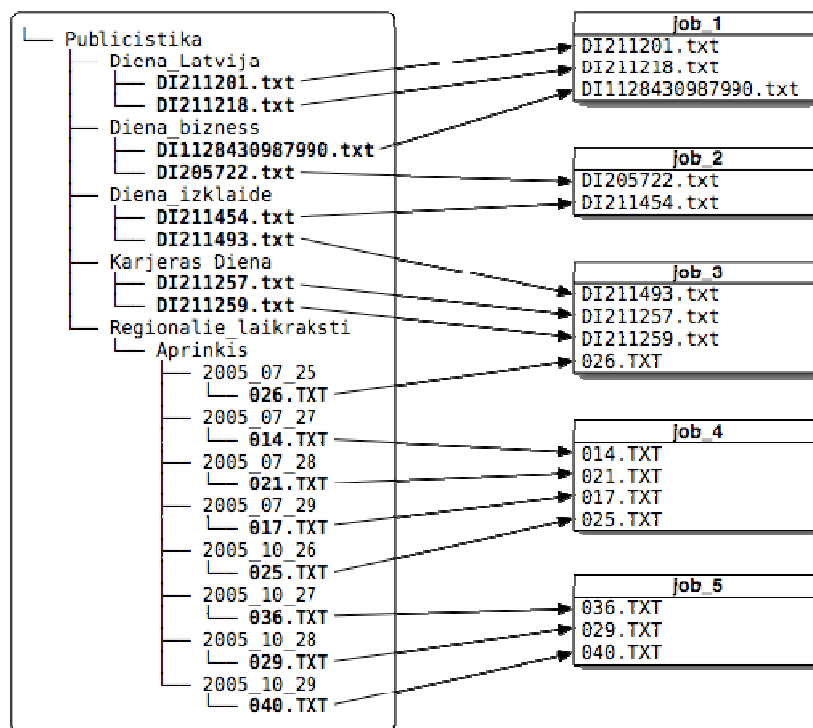
3.3.1. Uzdevuma sadalīšana

SemTi-Kamols sistēma apstrādā datnes pa vienai, bet apstrādājamo datņu ir ļoti daudz. Vienkāršākais veids, kā paralelizēt korpusa apstrādi būtu katrai ieejas datnei izveidot atsevišķu grid uzdevumu. Bet šī pieeja ir ļoti neefektīva mazām datnēm, kuru apstrāde aizņem mazāk laika nekā grid uzdevuma izpildes cikla pārējie posmi. Tāpēc ir izdevīgi vienā grid uzdevumā apvienot vairāku datņu apstrādi, tādējādi samazinot grid infrastruktūras radīto pārtēriņu.

Katras datnes apstrādes laiks ir ļoti atkarīgs no tajā esošo teikumu garuma, bet no eksperimentāliem rezultātiem² secināms, ka 256KB ieejas datņu tiek apstrādātas aptuveni 3 stundu laikā. Turpretī korpusā esošo datņu izmēri svārstās no aptuveni 1KB līdz 20KB, lai gan gadās arī lielākas. *Kamols* uzdevuma tipam ir parametrs, kas nosaka katram Grid uzdevumam *nosūtāmo ieejas datņu kopējo izmēru*.

¹ www.semti-kamols.lv

² "Harvesting national language text corpora from the Web", J.Džeriņš, K.Džonsons, "Human Language Technologies", 4-5 October, 2007, Kaunas, Lithuania.



2. attēls: Hipotētisks datņu sadalījums pa grid darbiem

Katram Kamols Grid darbam *izpildes mapē* tiek veidota *darba mape* ar nosaukumu `job_N`, kur `N` ir katram darbam piekārtots kārtas numurs. Šajā mapē tiek turēti katra darba izpildei nepieciešamās datnes.

Veidojot katru nākamo grid darbu, no *ieejas datņu mapes* pēc kārtas tiek ņemtas datnes tā, lai to kopējais izmērs nepārsniedz vēlamo. Kad nepieciešamais datņu daudzums ir savākts, tiek izveidots arhīvs `inputs.tgz`, kas satur visas atlasītās datnes.

Pēc šablona tiek izveidota arī grid uzdevuma izpildes scenārija datne `semTi-kamols.sh`, kurā tiek ierakstīts tikko izveidotā apstrādājamā ieejas datņu arhīva faila nosaukums. Šīs abas datnes tiek ierakstītas jaunajam grid uzdevumam izveidotajā *darba mapē*.

3.3.2. Darbu izpilde

Darba izpilde notiek, vadoties pēc scenārija, kas glabājas `semTi-kamols.sh` datnē. Kā jau minēts iepriekš, šī datne pēc šablona tiek veidota katram darbam, un satur visu nepieciešamo katram darbam individuālo informāciju.

Katra uzdevuma izpildei nepieciešamas divas lietas: 1) SemTi-Kamols lietotne un 2) apstrādājamās datnes. SemTi-Kamols lietotne visiem grid uzdevumiem ir viena un tā pati, un tā nav jāšūta līdz katram darbam. Ideālā variantā, kad SemTi-Kamols lietotne būs pabeigta, to varētu uzinstalēt uz visiem izpildes mezgliem. Bet pašlaik tā tiek nemitīgi uzlabota, un tās regulāra pārinstalēšana būtu nevajadzīga resursu izšķiešana. Tā vietā šī lietotne pēc katras jaunās versijas tiek augšuplādēta uz glabāšanas mezglu – katra grid darba izpildes sākumā tiek pārbaudīts, vai uz attiecīgā izpildes mezgla jau nav atrodamā šī lietotne. Ja lietotnes nav, tad tā tiek lejuplādēta, ja ir – nekā netiek lejuplādēts.

Pēc tam no glabāšanas mezglā tiek lejuplādēts un atarhivēts ieejas failu arhīvs. Cikliski visi tajā esošie faili tiek apstrādāti, katram no tiem palaižot SemTi-Kamols lietotni. Rezultātā tiek izveidots jauna arhīva datne `outputs.tgz`, kurā atrodas visi iegūtie HTML faili. Šī datne pēc darba veiksmīgas izpildes tiek iekopēta *darba mapē*.

3.3.3. Rezultātu apstrāde

Dokumenta tapšanas brīdī aktuālajā Grindes programmas versijā no Kamols darbu izpildes iespējams iegūt šādus rezultātus:

- Laikus, cik ilgi katrs uzdevums izpildījies uz izpildes mezglā. Šo informāciju var izmantot gan informatīviem mērķiem, gan arī lai varētu piemeklēt labāko darbiem *nosūtāmo ieejas datņu kopējo izmēru*. Šie laiki tiek saglabāti datnē `times.csv`.
- Analīzes rezultātu kopsavilkumu: cik vārdu pavisam ir katrā datnē, cik no tiem ir atpazīti, cik daudz vārdu savienojumu izdevies veiksmīgi atrast, un vidējais atrasto vārdu savienojumu garums. Šī informācija tiek saglabāta datnē `output_stats.csv`.

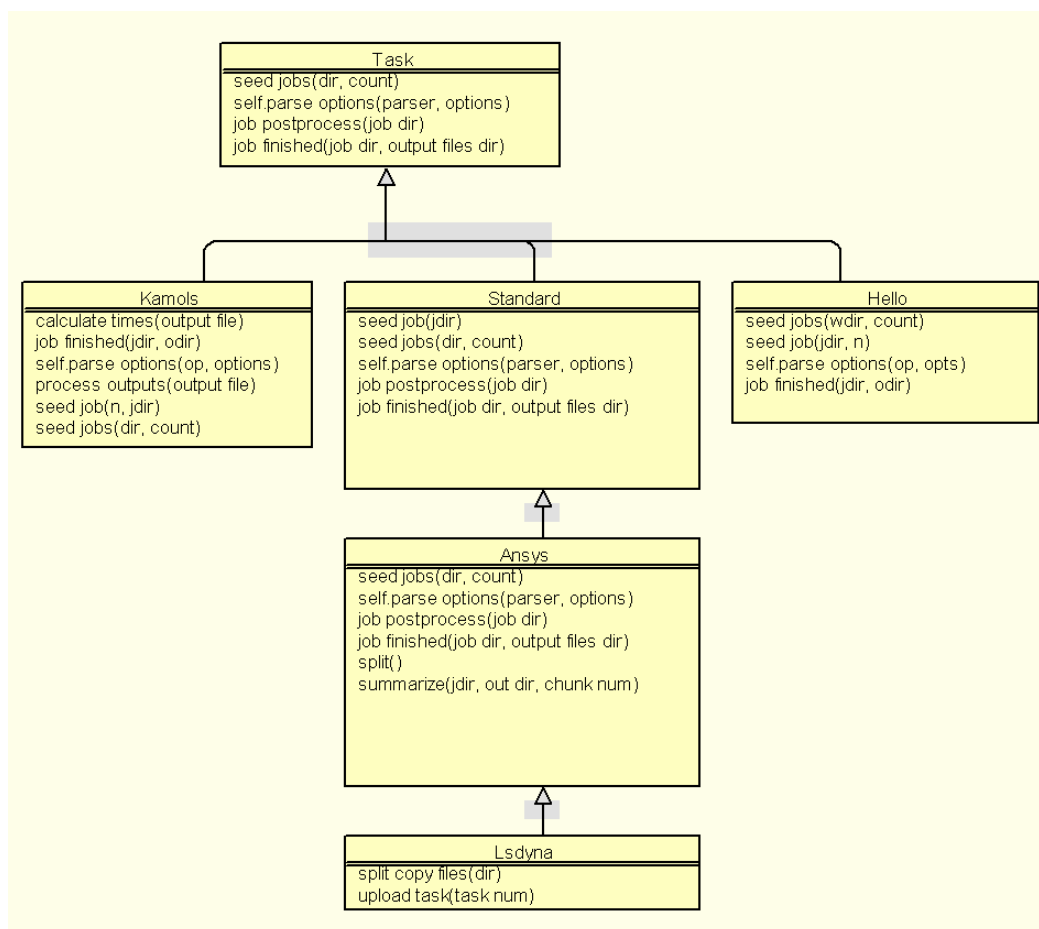
Rezultātu datnes netiek dzēstas, tādējādi iespējams arī iegūt visas izejas datnes. Šī darbība pašlaik vēl nav automatizēta, bet to var izdarīt, manuāli atarhivējot visas darbu rezultātu datnes.

3.4. GRINDER IZMANTOŠANA

Grinder palaišanai lieto komandu `grind`, kurai ir sekojoša sintakse:

```
grind <uzdevums> <operācija> <parametri_un_opcijas>*
```

uzdevums ir kāds no iepriekš definētām uzdevumu realizācijām. Pašlaik ir realizēti šādi uzdevumu tipi: `Kamols`, `Standard`, `ANSYS`, `Lsdyna` un `Hello`. Šo uzdevumu klašu hierarhija redzama 3. attēlā.



3. attēls Grinder uzdevumu klašu diagramma

Visiem uzdevumiem ir viena operācija `run`, kas realizē augstāk redzamo uzdevumu izpildes procesu. Uzdevumu klases var piedāvāt arī uzdevumiem specifiskas operācijas. Komandai `grind` ir arī palīdzības opcija `-h`, ar kuras palīdzību var iegūt īsu parametru un opciju aprakstu (skat. 4. attēlu).

Komanda `grind` ir lietotāja komanda, nevis fonā strādājošs process. Tāpēc, lai panāktu uzdevuma pilnīgu izpildi, iespējams rīkoties šādi:

- Kā paralēli izpildāmo uzdevumu skaitu norādīt pietiekami lielu skaitli. Ja reālo darbu skaits (pēc uzdevuma sadalīšanas darbos) būs mazāks vai vienāds ar šo skaitli, tad tie visi tiks nosūtīti izpildei. Vēlāk palaist komandu atkārtoti rezultātu saņemšanai un apstrādei.
- Ar rokām ik pa laikam palaist `grind` ar vieniem un tiem pašiem parametriem.
- Ievietot `grind` komandu ar visiem tās parametriem regulāri izpildāmo komandu sarakstā (izmantojot `cron` sistēmas komandu).

```
> grind --help
Usage: grinder <task> <operation> [options]

Options common to all tasks:
  -j, --job-count N           number of concurrently running jobs
                              (default 20)
  -n, --job-start-number N   number of the first job (if there are no
                              jobs in work directory; default 1)
  -w, --work-dir DIR         path to the working folder (default is
                              'grinder' in user's home directory; users
                              should not change any files in this folder)

Ansys job options:
  -u, --griduser STR         grid username - lfc directory for file
                              input/output
  --ansysin DIR              where user should put ansys tasks

Standard job options:
  -i, --indir DIR            where in*.zip files are taken from
  -o, --outdir DIR           where out*.zip files are put in

Hello job options:
  -m, --message STR          use this message instead of 'Hello world!'

Kamols specific options:
  -t, --texts-dir DIR        where input files are located
  -k, --chunk-size N         size of each chunk (in kilobytes,
                              default 512)
  -l, --keep-logs            keep log files
  --no-keep-logs             don't keep log files

Kamols specific operations:
  times                       calculates the job execution times and
                              outputs them to 'times.csv' in current
                              directory; for this operation to succeed
                              log files must be kept (the --keep-logs
                              option)
  process                     process job outputs and write them to
                              'output_stats.csv' file in current
                              directory

Other options:
  -h, --help                  show this message
```

4. attēls: grind komandas parametru iebūvētā dokumentācija

3.4.1. Opciju un parametru apraksts

Visiem uzdevumiem kopējie parametri:

--job-count N

Paralēli strādājošo darbu skaits. Noklusētā vērtība 20.

Ar šo parametru iespējams ierobežot palaižamo darbu skaitu. Tas vajadzīgs ja, piemēram, tiek izmantota programmatūra ar ierobežojumiem uz to izmantošanu. Vai arī ja nav pārliecības, ka uzdevumi vai to ieejas dati nav pilnīgi pareizi uzrakstīti un nav vērts sūtīt visus darbus, ja tie visi tiks izpildīti ar kļūdām.

--job-start-number N

Darbu numerācija sākas ar šo skaitli. Noklusētā vērtība 1.

Katram grid darbam tiek veidota atsevišķa mape. Šis skaitlis nosaka, ar kādu skaitli sākt numurēt šīs mapes. Šis skaitlis tiek izmantots tikai **uzsākot uzdevuma izpildi**. Ja *darba mapē* (skat. parametru `--work-dir`) jau ir izveidotas uzdevumu izpildes mapes, tad numerācija turpinās no lielākā izmantotā numura.

--work-dir MAPE

Uzdevuma darba mape. Grid darbu pagaidu mapes tiek veidotas šajā mapē.

Standard uzdevuma parametri:

--indir MAPE

Ieejas mape, kur atrodas uzdevumu .zip datnes.

--outdir MAPE

Rezultātu mape, kurā tiek novietotas katram darbam atbilstošās darba rezultātu .zip mapes.

ANSYS uzdevuma parametri:

--griduser VĀRDS

Uzdevuma datņu nogādāšana uz izpildes mezgliem notiek, izmantojot glabāšanas mezglu. VĀRDS ir mape glabāšanas mezglā, kur šīs datnes darbu izpildes laikā tiek glabātas.

--ANSYSin MAPE

Mape, kurā tiek meklēti ANSYS uzdevumu ieejas datnes.

Uzdevumu datnes no šejienes tiek paņemtas, un pēc tam notiek uzdevuma dalīšana darbos. Pēc darbu izpildes uzdevuma rezultāts iegūstams mapē, kas norādīta, izmantojot `--outdir` parametru.

SemTi-Kamols specifiskie parametri:

--texts-dir MAPE

Mape, kur atrodas apstrādājamās teksta datnes.

Visas datnes šajā mapē (un tās apakšmapēs) tiek uzskatītas par teksta datnēm, un tiek apstrādātas izmantojot SemTi-Kamols lietotni.

--chunk-size N

Nosūtāmo ieejas datņu kopējais izmērs, norādāms kilobaitos.

Katrā Grid darbā tiek apstrādātas vairākas teksta datnes, bet to kopējais apjoms nedrīkst pārsniegt N. Izņēmuma gadījums ir ja viena teksta datne pārsniedz šo izmēru – tādā gadījumā Grid darbā tiek apstrādāta šī viena datne.

--keep-logs

Vai saglabāt darba izpildes laikā rodušās žurnāla datnes.

Žurnāla datnēs iespējams iegūt informāciju par darbu izpildi. Šī opcija obligāti jānorāda, ja vēlāk tiek izmantota `times` operācija, jo datņu apstrādes laiki tiek fiksēti žurnāla failos.

SemTi-Kamols specifiskās operācijas:

times

Aprēķināt darbu izpildes laikus, rezultātus saglabājot datnē `times.csv`.

process

Analizēt darbu izpildes rezultātu datnes, datus saglabājot `output_stats.csv` datnē. Šādas datnes satūra fragments redzams 1. tabulā.

Job	File	Words	Recognized	Chunks	Avg.size
20024	Dailliteratura!Ermansons.Teicejs.txt.txt.html	126	100	73	1.73
20024	Dailliteratura!Bankovskis.Ofsors.txt.html	143	100	82	1.74
20022	Dailliteratura!Seglins.Kaktins.txt.txt.html	165	100	116	1.42
20022	Dailliteratura!Priede.Zemgales_bajariene_Riga.txt.html	161	100	83	1.94
20004	Publicistika!Diena_majai_un_gimenei!DI11097724478110.txt.html	302	84.77	105	2.07
20004	Publicistika!Diena_majai_un_gimenei!DI11098588851350.txt.html	335	94.63	103	2.91
20004	Publicistika!Diena_majai_un_gimenei!DI11256673632040.txt.html	1233	99.03	167	7.24
20004	Publicistika!Diena_majai_un_gimenei!DI10888601669890.txt.html	127	88.98	35	2.87
20004	Publicistika!Diena_majai_un_gimenei!DI211371.txt.html	272	95.96	37	6.77
20004	Publicistika!Karjeras_Diena!DI11096860837928.txt.html	1030	97.38	127	7.69
20004	Publicistika!Regionalie_laikraksti!Dzirkstele!117.TXT.html	1825	98.68	199	8.93
20004	Publicistika!Diena_majai_un_gimenei!DI11176320072531.txt.html	3785	99.97	404	9.36

1. tabula: SemTi-Kamols uzdevuma izpildes rezultāta faila fragments

4. SECINĀJUMI

Grinder sistēma veiksmīgi tiek lietota divu dažādu tipu uzdevumu risināšanai. Neskatoties uz to, vēl vairākas lietas sistēmā nepieciešams realizēt:

- Jāvar darbināt Grinder kā sistēmas procesu, lai lietotājiem tas nav jālaiž pašiem ar roku vai jāizmanto `cron` sistēma. Lai to panāktu, nepieciešams realizēt sistēmas un lietotāju konfigurācijas failu lasīšanu.
- Jārealizē e-pastu sūtīšana noteiktos brīžos, piemēram, uzsākot uzdevumu apstrādi, kļūdu gadījumā un beidzot uzdevumu apstrādi.
- Būtu vēlams realizēt sistēmas darbības kontroli (ieejas datņu augšupielādi, uzdevuma izpildes statusa vērošanu un rezultātu lejupielādi) izmantojot tīmekļa lietotni.