Performance-Test Bibsonomy2-REST-API

(1) Allgemeines

Test-Query	/api/users/dblp/posts?resourcetype=bibtex&start=0&end=[n]
Testwerte:	1000 x [n] = 1 1000 x [n] = 20 1000 x [n] = 100 1000 x [n] = 500 1000 x [n] = 1000 500 x [n] = 5000 500 x [n] = 10000 ➔ 6000 Anfragen
Anzahl Testclients:	1, 3, 5, 10, 15, 22 ➔ 330.000 Anfragen
Erfasste Testparameter	- System Load (1 minute average load) [SYS_LOAD] - Prozent freier Hauptspeicher [RAM_FREE] - Prozent freier SWAP-Space [SWAP_FREE] - Zeit vom Absenden der Anfrage beim Client bis zum vollstaendigen Erhalt der Antwort [ELAPSED]
RAM size of gromit	8179812k

(2)Min|Max|Avg|StdDev SYS_LOAD

Format: *Min | Max*
Avg
StdDev

clients / posts	***1***	***3***	***5***	***10***	***15***	***22***
1	*0.09	1.23* ***0.65*** *0.33*	*1.29	2.82* ***2.17*** *0.42*	*1.14	3.49* ***2.40*** *0.71*	*1.12	4.93* ***2.99*** *0.95*	*0.97	5.96* ***3.79*** *1.35*	*1.1	5.81* ***4.04*** *0.95*
20	*1.23	1.9* ***1.65*** *0.24*	*2.59	3.19* ***2.96*** *0.15*	*2.29	3.24* ***2.61*** *0.23*	*2.53	4.93* ***4.08*** *0.6*	*2.08	4.7* ***2.54*** *0.55*	*3.57	5.62* ***4.21*** *0.36*
100	*1.9	2.2* ***2.04*** *0.07*	*2.96	3.46* ***3.14*** *0.14*	*3.08	4.26* ***3.63*** *0.21*	*3.5	5.38* ***4.39*** *0.30*	*2.69	5.57* ***4.36*** *0.44*	*3.54	5.3* ***4.21*** *0.34*
500	*2	2.23* ***2.1*** *0.06*	*2.96	4.41* ***3.74*** *0.42*	*3.62	5.11* ***4.64*** *0.23*	*3.5	5.81* ***4.79*** *0.45*	*4.04	5.68* ***5*** *0.27*	*3.99	6.09* ***4.86*** *0.27*
1000	*1.64	2.33* ***1.94*** *0.18*	*3.51	4.53* ***4.01*** *0.231*	*4.45	6.32* ***4.92*** *0.25*	*3.8	6.65* ***4.92*** *0.51*	*4.1	6.13* ***5.06*** *0.4*	*4.09	6.29* ***5.12*** *0.34*
5000	*1.79	2.45* ***2.12*** *0.15*	*3.71	4.83* ***4.27*** *0.25*	*4.48	7.04* ***5.63*** *0.56*	*4.24	6.96* ***5.32*** *0.53*	*4.1	6.57* ***5.21*** *0.41*	*4.09	6.54* ***5.45*** *0.34*
10000	*1.8	2.63* ***2.15*** *0.13*	*3.94	7.6* ***5.14*** *0.56*	*2.65	7.35* ***5.42*** *0.56*	*2	6.96* ***5.14*** *0.88*	*2.6	6.8* ***5.26*** *0.47*	*2.3	7.14* ***5.59*** *0.49*

```
select substring(query,-5) as posts, num_parallel as np, min(sys_load) as min, max(sys_load) as
max, round(avg(sys_load),2) as avg, round(stddev(sys_load),2) as dev from results group by
num_parallel, query order by posts, num_parallel;
```

(3)Min|Max|Avg RAM_FREE

Format: *Min | Max*
Avg
(Standardabweichung war fast konstant 0)

Clients / posts	***1***	***3***	***5***	***10***	***15***	***22***
1	*0.02 | 0.03* ***0.02***	*0.02 | 0.03* ***0.03***	*0.63 | 0.64* ***0.63***	*0.39 | 0.41* ***0.4***	*0.33 | 0.34* ***0.33***	*0.33 | 0.34* ***0.33***
20	*0.03 | 0.03* ***0.03***	*0.02 | 0.03* ***0.03***	*0.63 | 0.63* ***0.63***	*0.39 | 0.40* ***0.39***	*0.32 | 0.33* ***0.33***	*0.30 | 0.30* ***0.30***
100	*0.03 | 0.03* ***0.03***	*0.03 | 0.03* ***0.03***	*0.63 | 0.63* ***0.63***	*0.39 | 0.40* ***0.39***	*0.32 | 0.33* ***0.32***	*0.30 | 0.30* ***0.30***
500	*0.03 | 0.03* ***0.03***	*0.01 | 0.02* ***0.02***	*0.62 | 0.63* ***0.63***	*0.38 | 0.39* ***0.39***	*0.30 | 0.33* ***0.32***	*0.30 | 0.30* ***0.30***
1000	*0.02 | 0.03* ***0.02***	*0.01 | 0.02* ***0.02***	*0.58 | 0.63* ***0.6***	*0.35 | 0.40* ***0.3***	*0.31 | 0.33* ***0.32***	*0.28 | 0.31* ***0.30***
5000	*0 | 0.02* ***0.01***	*0 | 0.03* ***0.02***	*0.45 | 0.62* ***0.52***	*0.35 | 0.40* ***0.38***	*0.28 | 0.33* ***0.3***	*0.27 | 0.31* ***0.29***
10000	*0.01 | 0.03* ***0.02***	*0 | 0.04* ***0.02***	*0.35 | 0.48* ***0.42***	*0.35 | 0.39* ***0.37***	*0.28 | 0.31* ***0.29***	*0.26 | 0.30* ***0.28***

select substring(query,-5) as posts, num_parallel as np, round(min(ram_free / 8179812),2) as min, round(max(ram_free / 8179812),2) as max, round(avg(ram_free / 8179812),2) as avg, round(stddev(ram_free / 8179812),2) as dev from results group by num_parallel, query order by posts, num_parallel;

(4)Hat Swapping stattgefunden?

Maximale Anzahl der ausgelagerten Kilobytes

Clients / posts	***1***	***3***	***5***	***10***	***15***	***22***
1	*0*	*0*	*0*	*0*	*0*	*0*
20	*0*	*0*	*0*	*0*	*0*	*0*
100	*0*	*0*	*0*	*0*	*0*	*0*
500	*0*	*0*	*0*	*0*	*0*	*0*
1000	*0*	*0*	*0*	*0*	*0*	*0*
5000	*0*	*0*	*0*	*0*	*0*	*0*
10000	*0*	***15136***	*0*	*0*	*0*	*0*

(5)Min/Max/Avg/StdDev ELAPSED

(Format: s.o.)

clients ***posts***	***1***	***3***	***5***	***10***	***15***	***22***
1	*0.029 | 1.267* ***0.043*** *0.115*	*0.030 | 1.338* ***0.079*** *0.203*	*0.025 | 4.657* ***0.113*** *0.290*	*0.026 | 5.176* ***0.169*** *0.344*	*0.024 | 2.230* ***0.345*** *0.484*	*0.028 | 6.329* ***0.516*** *0.690*
20	*0.034 | 1.275* ***0.047*** *0.115*	*0.035 | 1.313* ***0.083*** *0.200*	*0.031 | 1.380* ***0.116*** *0.261*	*0.030 | 2.310* ***0.238*** *0.380*	*0.031 | 2.883* ***0.369*** *0.509*	*0.034 | 5.624* ***0.569*** *0.756*
100	*0.053 | 1.477* ***0.080*** *0.130*	*0.057 | 1.511* ***0.120*** *0.207*	*0.052 | 1.594* ***0.161*** *0.263*	*0.053 | 4.103* ***0.367*** *0.472*	*0.052 | 7.474* ***0.516*** *0.666*	*0.055 | 9.772* ***0.777*** *0.989*
500	*0.150 | 1.398* ***0.183*** *0.126*	*0.155 | 1.703* ***0.245*** *0.214*	*0.152 | 3.569* ***0.345*** *0.294*	*0.153 | 5.977* ***0.682*** *0.706*	*0.150 | 10.557* ***1.068*** *1.221*	*0.153 | 13.644* ***1.493*** *1.757*
1000	*0.272 | 1.736* ***0.348*** *0.144*	*0.283 | 1.878* ***0.408*** *0.223*	*0.271 | 5.607* ***0.600*** *0.389*	*0.279 | 16.541* ***1.578*** *1.697*	*0.278 | 16.158* ***1.885*** *1.978*	*0.273 | 17.272* ***2.504*** *2.452*
5000	*1.240 | 2.963* ***1.342*** *0.161*	*1.339 | 3.326* ***1.631*** *0.277*	*1.356 | 30.875|* ***2.552*** *1.124*	1.344 | 15.652 **4.266** 2.519	*1.367 | 17.829* ***4.940*** *2.648*	*1.319 | 19.373* ***5.378*** *2.562*
10000	*2.484 | 4.264* ***2.592*** *0.163*	*2.813 | 36.713* ***3.742*** *1.644*	*2.468 | 19.608* ***5.293*** *1.774*	*2.425 | 17.198* ***5.797*** *2.410*	*2.428 | 17.951* ***6.312*** *29*	*2.425 | 19.692* ***6.577*** *1.820*

select substring(query,-5) as posts, num_parallel as np, round(min(elapsed),3) as min, round(max(elapsed),3) as max, round(avg(elapsed),3) as avg, round(stddev(elapsed),3) as dev from results group by num_parallel, query order by posts, num_parallel;

(6)Max/Avg number of requests completed per second (measured on client side)

Format: Min | **Avg**

1 ***clients*** ***posts***	***1***	***3***	***5***	***10***	***15***	***22***
1	*32 |* ***22.73***	*70 |* ***37.04***	*97 |* ***48.54***	*97 |* ***27.86***	*94 |* ***45.87***	*95 |* ***42.77***
20	*28 |* ***21.74***	*61 |* ***33.33***	*86 |* ***43.4***	*63 |* ***20.24***	*89 |* ***40.54***	*79 |* ***36.52***
100	*18 |* ***12.35***	*39 |* ***23.62***	*53 |* ***28.90***	*47 |* ***13.39***	*53 |* ***22.76***	*49 |* ***22.90***
500	*7 |* ***5.46***	*17 |* ***12***	*21 |* ***12.47***	*19 |* ***7.87***	*21 |* ***11.38***	*21 |* ***11.69***
1000	*21 |* ***11.69***	*10 |* ***7.32***	*12 |* ***6.78***	12 | **4.16**	*13 |* ***6.55***	*13 |* ***6.55***
5000	*1 |* ***1***	*3 |* ***1.86***	*4 |* ***1.83***	5 | **1.54**	*9 |* ***2.39***	*13 |* ***2.95***
10000	*1 |* ***1***	*3 |* ***1.12***	*3 |* ***1.32***	*6 |* ***1.42***	*11 |* ***2.13***	*14 |* ***2.69***

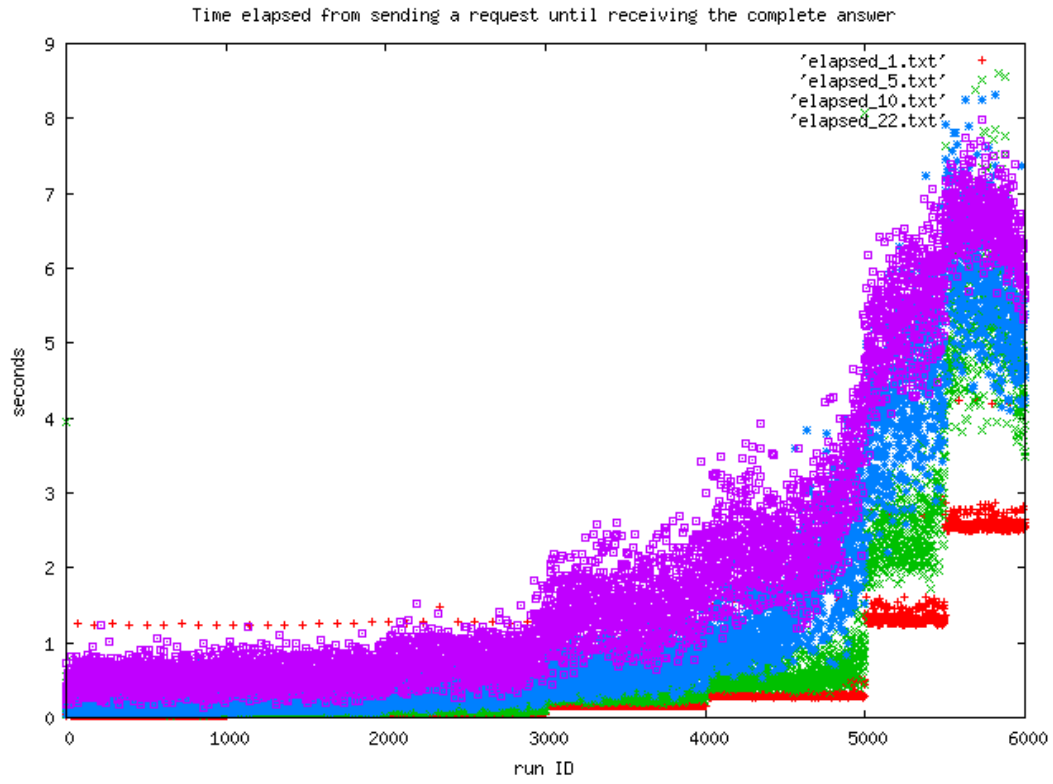
select distinct R.np, R.posts, round(min(R.req_per_sec),2) as min, round(max(R.req_per_sec),2) as max, round(avg(R.req_per_sec),2) as avg from (select timestamp, num_parallel as np, substring(query, -5) as posts, count(*) as req_per_sec from results group by timestamp, num_parallel, query) R group by R.np, R.posts order by R.posts, R.np;

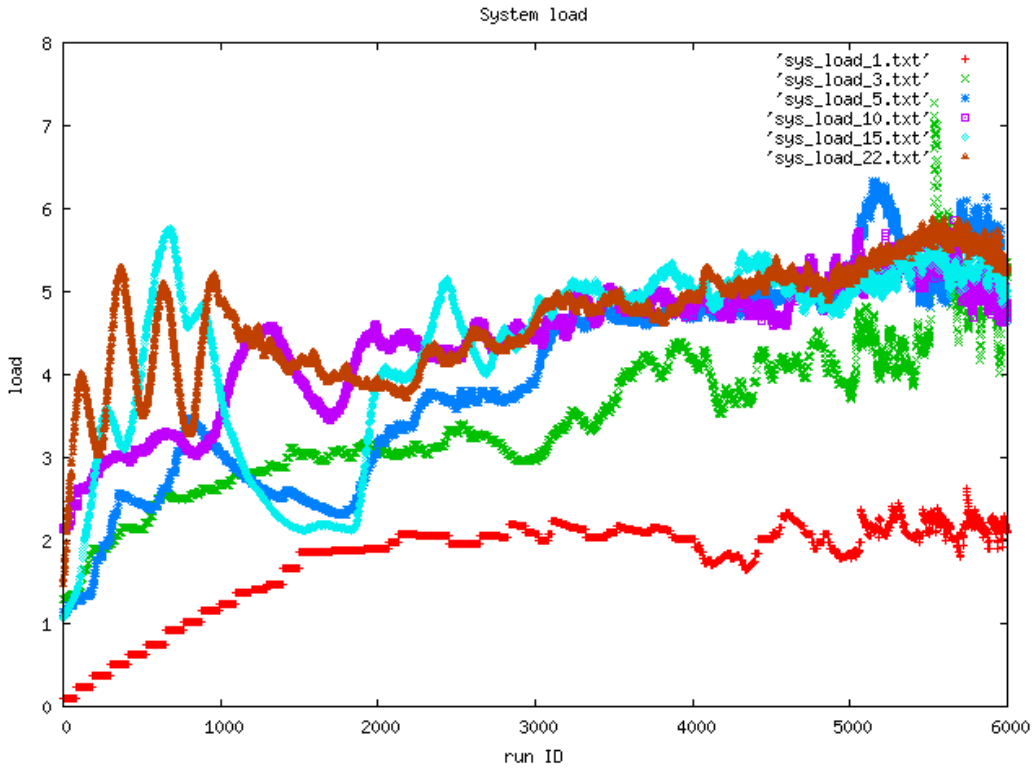
(7)Number of Server Errors (HTTP status 500)

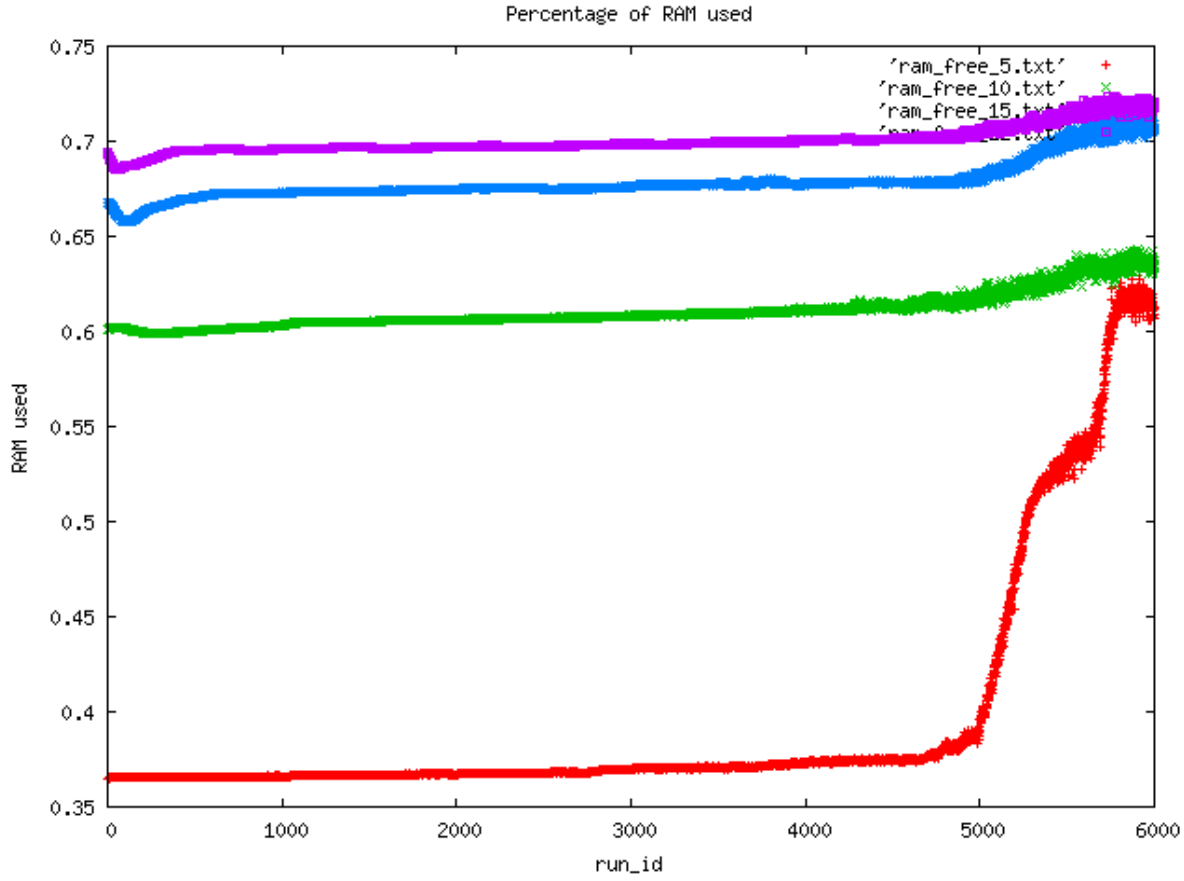
clients / ***posts***	***1***	***3***	***5***	***10***	***15***	***22***
1	*0*	*0*	*0*	*0*	*0*	*0*
20	*0*	*0*	*0*	*0*	*0*	*0*
100	*0*	*0*	*0*	*0*	***1***	***10***
500	*0*	*0*	*0*	*0*	***32***	***427***
1000	*0*	*0*	*0*	**202**	***581***	***2325***
5000	*0*	*0*	***1***	***771***	***2648***	***5320***
10000	*0*	***1***	***21***	***1127***	***3971***	***6861***

select substring(query, -5) as posts, num_parallel as np, count(*) from results where http_status=500 group by query, num_parallel order by query, num_parallel

(8)Graphical statistics




System load
'sys_load_1.txt'
'sys_load_3.txt'
'sys_load_5.txt'
'sys_load_10.txt'
'sys_load_15.txt'
'sys_load_22.txt'
load
run ID
0
1
2
3
4
5
6
7
8
0
1000
2000
3000
4000
5000
6000


Percentage of RAM used
'ram_free_5.txt'
'ram_free_10.txt'
'ram_free_15.txt'
RAM used
run_id
0.35
0.4
0.45
0.5
0.55
0.6
0.65
0.7
0.75
0
1000
2000
3000
4000
5000
6000