

TP 2 - Recuperação de Informação

Yuri Diego Santos Niitsuma

O Trabalho consiste em criar um índice invertido para a coleção do TP1.

Este trabalho foi dividido em 3 passos.

- A primeira parte `pass1` consiste em criar um vocabulário guardando as strings dos termos no arquivo `terms.dump` mais as informações de (tamanho, posição) no arquivo em outro arquivo `hash_table.tbl`. Ao mesmo tempo que é criado um arquivo de mapeamento (posição/ponteiro) de cada i-ésimo documento no arquivo `[nome_do_arquivo_da_colecao].index` e as listas de URLs em `[nome_do_arquivo_da_colecao].urlist`.
Os termos são guardados temporariamente em vários arquivos na pasta `terms` que será descrito na seção seguinte.
- A segunda parte `pass2` faz a ordenação e a união dos arquivos na pasta `terms` em um único arquivo `terms.index` (já ordenado). Nesta etapa o arquivo `hash_table.tbl` também é ordenado em ordem alfabética dos termos para uma busca binária.
- A terceira parte `pass3` é o programa que faz as consultas dos termos no índice invertido. Digitando `/help` lista algumas funções disponíveis. Para efetuar uma busca basta digitar os termos separados por espaços. O método utilizado é a busca booleana.
Algumas funções disponíveis são:
 - `/list` : lista todos os termos junto com sua frequência totais
 - `/s <termo> OU /seach <termo> : ???`
 - `/quit` : fecha o programa

Execução

Compila em 3 binários `pass1.out`, `pass2.out` e `pass3.out`.

```
$ make
```

Apaga todos os arquivos criados para o índice invertido e os binários executáveis.

```
$ make clean
```

Executa os 3 passos no arquivo `html_pages.txt` (estimado 16 minutos utilizando um SSD)

```
$ make test
```

Executa os 3 passos no arquivo `ri_2018_collection.dat` (não faça isso)

```
$ make run
```

Caso deseje evitar repetir os dois primeiros passos e executar diretamente o passo 3. Execute

```
$ ./pass3.out <nome_do_arquivo_da_colecao>
```

Estrutura dos arquivos

Todos os arquivos são armazenados no formato Endianness da máquina rodada. Nos exemplos seguintes os dados são descritos em Little Endian.

[nome_do_arquivo].urllist

Arquivo contendo lista de URLs coletadas e separados por um '\n'. Será útil no próximo trabalho prático.

[nome_do_arquivo].index

Arquivo que mantém informações das posições da URL do no arquivo `[nome_do_arquivo].urllist` e a posição do conteúdo do HTML no arquivo da coleção.

pointer to URL	pointer to HTML content
00 00 00 00 00 00 00 00	10 00 00 00 00 00 00 00
0D 00 00 00 00 00 00 00	61 2B 05 00 00 00 00 00
...	...

Os ponteiros apontam para a posição no arquivo da coleção utilizada no TP1.

```
|||github.com|<HTML> XXX </HTML>|||youtube.com|<HTML> XXX </HTML>|||  
...  
showshow.com|<HTML> XXX </HTML>|||
```

O i-ésimo documento é indicado ordenadamente neste arquivo. Iniciando do 0.

hash_table.tbl

Estrutura serve pra mapear o ID (**hash_id**) e a posição (**position**) da string do termo com tamanho **chars_length** no arquivo **terms.dump**.

O **hash_id** é uma identificação única para cada termo.

```
struct HashBlock {  
    size_t hash_id; // unique id of term  
    size_t position; // position to char dump  
    size_t chars_length; // length of term  
    size_t pointer_to_term; // ponteiro para o block do termo  
    size_t freq; // freq of term on all documents  
};
```











Exemplo antes da ordenação

hash_id	position	chars_length	pointer_to_term	freq
0	0	3	0	50
1	3	10	50	11
2	13	6	...	13
...
n	position(n)	size(n)	...	5

Exemplo após a ordenação, observe que os **hash_ids** (8 bytes da esquerda) estão aparentemente, aleatórios. Como descrito anteriormente, a ordenação foi alfabética consultando o `terms.dump`.

/home/yuri/src/ufmg/2018_1_ri/tp2/hash_table.tbl - Bless

File Edit View Search Tools Help



hash_table.tbl x

00000000	1E	D6	00	00	00	00	00	00	36	C8	06	00	00	00	00	00	01	00	00	00	00	00	00	F3	21	12	00	00	00	00	00	01	00	00	00	00	00	00	
00000028	8A	01	00	00	00	00	00	00	EB	09	00	00	00	00	00	00	01	00	00	00	00	00	00	09	25	06	00	00	00	00	00	87	05	00	00	00	00	00	
00000050	02	07	00	00	00	00	00	00	C2	2E	00	00	00	00	00	00	02	00	00	00	00	00	00	F3	41	09	00	00	00	00	00	BF	06	00	00	00	00	00	
00000078	43	12	00	00	00	00	00	00	51	80	00	00	00	00	00	00	03	00	00	00	00	00	00	04	87	0C	00	00	00	00	00	CB	00	00	00	00	00	00	
000000a0	71	E0	00	00	00	00	00	00	D7	26	07	00	00	00	00	00	04	00	00	00	00	00	00	55	31	12	00	00	00	00	00	02	00	00	00	00	00	00	
000000c8	80	E4	00	00	00	00	00	00	3B	47	07	00	00	00	00	00	0A	00	00	00	00	00	00	9A	39	12	00	00	00	00	00	01	00	00	00	00	00	00	
0000000f	7F	E4	00	00	00	00	00	00	31	47	07	00	00	00	00	00	0A	00	00	00	00	00	00	99	39	12	00	00	00	00	00	01	00	00	00	00	00	00	
00000118	96	33	00	00	00	00	00	00	6E	7B	01	00	00	00	00	00	08	00	00	00	00	00	00	04	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000140	95	33	00	00	00	00	00	00	66	7B	01	00	00	00	00	00	08	00	00	00	00	00	00	03	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000168	78	33	00	00	00	00	00	00	AD	7A	01	00	00	00	00	00	08	00	00	00	00	00	00	84	4D	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000190	D6	33	00	00	00	00	00	00	0E	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	A9	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
000001b8	DA	33	00	00	00	00	00	00	2A	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	AD	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
000001e0	D2	33	00	00	00	00	00	00	F2	7C	01	00	00	00	00	00	07	00	00	00	00	00	00	A4	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000208	88	33	00	00	00	00	00	00	12	7B	01	00	00	00	00	00	07	00	00	00	00	00	00	00	A7	4D	0F	00	00	00	00	00	01	00	00	00	00	00	00
00000230	10	1B	00	00	00	00	00	00	F2	BF	00	00	00	00	00	00	04	00	00	00	00	00	00	9A	98	0D	00	00	00	00	00	49	00	00	00	00	00	00	
00000258	0F	33	00	00	00	00	00	00	7B	77	01	00	00	00	00	00	07	00	00	00	00	00	00	54	4B	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000280	DE	33	00	00	00	00	00	00	46	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	B1	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
000002a8	58	33	00	00	00	00	00	00	E7	79	01	00	00	00	00	00	07	00	00	00	00	00	00	91	4C	0F	00	00	00	00	00	01	00	00	00	00	00	00	
000002d0	8A	33	00	00	00	00	00	00	20	7B	01	00	00	00	00	00	07	00	00	00	00	00	00	A9	4D	0F	00	00	00	00	00	01	00	00	00	00	00	00	
0000028f	DD	33	00	00	00	00	00	00	3F	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	B0	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000320	B9	33	00	00	00	00	00	00	4F	7C	01	00	00	00	00	00	07	00	00	00	00	00	00	4C	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000348	D5	33	00	00	00	00	00	00	07	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	A7	4E	0F	00	00	00	00	00	02	00	00	00	00	00	00	
00000370	D1	33	00	00	00	00	00	00	EB	7C	01	00	00	00	00	00	07	00	00	00	00	00	00	A3	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000398	D9	33	00	00	00	00	00	00	23	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	AC	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
000003c0	87	33	00	00	00	00	00	00	0B	7B	01	00	00	00	00	00	07	00	00	00	00	00	00	A6	4D	0F	00	00	00	00	00	01	00	00	00	00	00	00	
000003e8	D8	33	00	00	00	00	00	00	1C	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	AB	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000410	A7	34	00	00	00	00	00	00	67	83	01	00	00	00	00	00	06	00	00	00	00	00	00	64	59	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000438	D4	33	00	00	00	00	00	00	00	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	A6	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000460	BB	33	00	00	00	00	00	00	5D	7C	01	00	00	00	00	00	07	00	00	00	00	00	00	4E	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000488	5A	33	00	00	00	00	00	00	F5	79	01	00	00	00	00	00	07	00	00	00	00	00	00	93	4C	0F	00	00	00	00	00	01	00	00	00	00	00	00	
000004b0	08	33	00	00	00	00	00	00	4F	77	01	00	00	00	00	00	07	00	00	00	00	00	00	AE	4A	0F	00	00	00	00	00	01	00	00	00	00	00	00	
000004d8	8E	33	00	00	00	00	00	00	3C	7B	01	00	00	00	00	00	07	00	00	00	00	00	00	B1	4D	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000500	6F	33	00	00	00	00	00	00	77	7A	01	00	00	00	00	00	07	00	00	00	00	00	00	5F	4D	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000528	A3	33	00	00	00	00	00	00	C1	7B	01	00	00	00	00	00	07	00	00	00	00	00	00	23	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000550	D7	33	00	00	00	00	00	00	15	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	AA	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
00000578	DC	33	00	00	00	00	00	00	38	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	AF	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
000005a0	DB	33	00	00	00	00	00	00	31	7D	01	00	00	00	00	00	07	00	00	00	00	00	00	AE	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
000005c8	D0	33	00	00	00	00	00	00	E4	7C	01	00	00	00	00	00	07	00	00	00	00	00	00	A2	4E	0F	00	00	00	00	00	01	00	00	00	00	00	00	
0000050f	89	33	00	00	00	00	00	00	19	7B	01	00	00	00	00	00	07	00	00	00	00																		

terms.dump

Contém strings concatenadas para consulta utilizando parâmetros do arquivo **hash_table.tbl**.

<term1><term2><term3>...<termN>

```

File Edit View Search Tools Help
terms.dump x
00000000 67 31 6F 70 6F 72 74 61 6C 64 65 6E 6F 74 EF gloportal denot.
0000000f BF BD 63 69 61 73 64 61 67 6C 6F 62 6F 65 64 ..ciadagloboed
0000001e 69 74 6F 72 69 61 73 61 67 72 6F 63 61 72 72 itoriasagro carr
0000002d 6F 73 63 69 EF BF BD 6E 63 69 61 65 73 61 EF osci...nciaesa.
0000003c BF BD 64 65 63 6F 6E 63 75 72 73 6F 73 65 6D ..deconcurso sem
0000004b 70 72 65 67 6F EF BF BD 6F 75 6E EF BF BD 6F prego...oun...o
0000005a 65 63 6F 6E 6F 6D 69 61 65 64 75 63 61 EF BF economia educa..
00000069 BD EF BF BD 6F 65 73 70 6F 72 74 65 6D 75 6E ....oesporte mun
00000078 64 6F 6D EF BF BD 73 69 63 61 6E 61 74 75 72 dom...sicanatur
00000087 65 7A 61 70 6C 61 6E 65 74 61 62 69 7A 61 72 ezaplaneta bizar
00000096 72 6F 70 6F 6C EF BF BD 74 69 63 61 70 6F 70 ropol...ticapop
000000a5 61 72 74 65 74 65 63 6E 6F 6C 6F 67 69 61 67 artetecnologiag
000000b4 61 6D 65 73 74 75 72 69 73 6D 6F 76 69 61 67 amesturismo viag
000000c3 65 6D 74 6F 64 61 73 61 73 72 65 67 69 EF BF emtodasas regi..
000000d2 BD 65 73 63 65 6E 74 72 6F 6F 65 73 74 65 64 .escentrooested
000000e1 69 73 74 72 69 74 6F 66 65 64 65 72 61 6C 67 istrito federalg
000000f 6F 69 EF BF BD 73 6D 61 74 6F 67 72 6F 73 73 oi...smatogross
000000 6F 64 6F 73 75 6C 6E 6F 72 64 65 73 74 65 61 odosulnordestea
0000010e 6C 61 67 6F 61 73 62 61 68 69 61 63 65 61 72 lagoas bahia ce ar
0000011d EF BF BD 6D 61 72 61 6E 68 EF BF BD 6F 70 61 ...maranh...opa
0000012c 72 61 EF BF BD 62 61 70 65 72 6E 61 6D 62 75 ra...bapernambu
0000013b 63 6F 72 65 63 69 66 65 72 65 67 69 EF BF BD coreciferegi...
0000014a 6F 63 61 72 75 61 72 75 70 65 74 72 6F 6C 69 ocaruarupetroli
00000159 6E 61 70 69 61 75 EF BF BD 72 69 6F 67 72 61 napiau...riogra
00000168 6E 64 65 6E 6F 72 74 65 73 65 72 67 69 70 65 ndenortesergipe
00000177 61 63 72 65 62 72 61 6E 63 6F 63 72 75 7A 65 acrebrancocruze
00000186 69 72 6F 61 6D 61 70 EF BF BD 61 6D 61 7A 6F iroamap...amazo
00000195 6E 61 73 70 61 72 EF BF BD 62 65 6C EF BF BD naspar...bel...
000001a4 6D 73 61 6E 74 61 72 EF BF BD 6D 72 6F 6E 64 msantar...mrond
000001b3 EF BF BD 6E 69 61 61 72 69 71 75 65 6D 65 73 ...niaariquesmes
000001c2 76 61 6C 65 6A 61 6D 61 72 69 63 61 63 6F 61 valejamaricacoa
000001d1 6C 7A 6F 6E 61 6D 61 74 61 6A 69 70 61 72 61 lzonamatajipara
000001e0 6E EF BF BD 63 65 6E 74 72 61 6C 70 6F 72 74 n...centralport
Offset: 0x0 / 0x79c61 Selection: None INS

```

coleção de termos na pasta terms

Esta pasta serve para armazenar a frequência e posição do termo no HTML do arquivo da coleção (não é a posição na coleção).

Ele segue a estrutura:

```
struct {  
    size_t hash_id;  
    size_t document_id;  
    size_t position;  
};
```

hash_id	document_id	position
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	B1 09 00 00 00 00 00 00
01 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	B6 09 00 00 00 00 00 00
...

Cada termo tem um ID único (**hash_id**) para evitar criar um arquivo e evitar *paddings* dos arquivos no sistema operacional. Para cada termo é dividido em blocos na pasta termos decidido pelo.

$(\text{hash_id} / \text{MANY_ON_DAT_BLOCK})$

MANY_ON_DAT_BLOCK

MANY_ON_DAT_BLOCK é definido no **term_manage.h**.

Os primeiros MANY_ON_DAT_BLOCK termos são armazenados no arquivo **0**.

Os próximos MANY_ON_DAT_BLOCK no **1** e assim por diante.

Observe que como a varredura segue sequencialmente nos documentos, o document_id será ordenado nestes arquivos.

Após o primeiro passo, é feito a ordenação em cada arquivo temporário e depois é feito a junção em um único arquivo `terms.dump`.

A ordenação utilizada nestes arquivos é feito utilizando MergeSort externo iterativo.


```
/home/yuri/src/ufmg/2018_1_ri/tp2/terms.index - Bless
File Edit View Search Tools Help
terms.index x
00000000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 B1 09 00 00 00 00 00 00
00000018 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2E 3B 00 00 00 00 00 00
00000030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 74 B4 00 00 00 00 00 00
00000048 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 52 DE 00 00 00 00 00 00
00000060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 68 DF 00 00 00 00 00 00
00000078 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 D2 EB 00 00 00 00 00 00
00000090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 D9 A0 03 00 00 00 00 00
000000a8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 B3 60 04 00 00 00 00 00
000000c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 14 67 04 00 00 00 00 00
000000d8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 1D A3 04 00 00 00 00 00
000000f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 5E B6 04 00 00 00 00 00
00000108 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 BD 04 00 00 00 00 00
00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 E7 BE 04 00 00 00 00 00
00000138 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 52 0F 05 00 00 00 00 00
00000150 00 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 F9 57 02 00 00 00 00 00
00000168 00 00 00 00 00 00 00 00 02 00 00 00 00 00 00 00 CA 67 02 00 00 00 00 00
00000180 00 00 00 00 00 00 00 00 07 00 00 00 00 00 00 00 9C 00 00 00 00 00 00 00
00000198 00 00 00 00 00 00 00 00 04 03 00 00 00 00 00 00 FE DF 01 00 00 00 00 00
000001b0 00 00 00 00 00 00 00 00 F5 03 00 00 00 00 00 00 F8 3E 02 00 00 00 00 00
000001c8 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 B6 09 00 00 00 00 00 00
000001e0 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 4A F4 00 00 00 00 00 00
0000018f 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 67 C3 01 00 00 00 00 00
00000210 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 8A F2 01 00 00 00 00 00
00000228 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2B F3 01 00 00 00 00 00
00000240 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 85 94 03 00 00 00 00 00
00000258 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 50 75 04 00 00 00 00 00
00000270 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2D 9A 04 00 00 00 00 00
00000288 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 13 A8 04 00 00 00 00 00
000002a0 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 A6 E2 04 00 00 00 00 00
000002b8 01 00 00 00 00 00 00 00 01 00 00 00 00 00 00 88 32 01 00 00 00 00 00
000002d0 01 00 00 00 00 00 00 00 01 00 00 00 00 00 00 53 3A 01 00 00 00 00 00
000002e8 01 00 00 00 00 00 00 00 01 00 00 00 00 00 00 2D 58 01 00 00 00 00 00
00000300 01 00 00 00 00 00 00 00 01 00 00 00 00 00 00 39 58 01 00 00 00 00 00
00000318 01 00 00 00 00 00 00 00 01 00 00 00 00 00 00 54 58 01 00 00 00 00 00
00000330 01 00 00 00 00 00 00 00 01 00 00 00 00 00 00 7C 5F 01 00 00 00 00 00
00000348 01 00 00 00 00 00 00 00 01 00 00 00 00 00 00 FF 6F 01 00 00 00 00 00
00000360 01 00 00 00 00 00 00 00 01 00 00 00 00 00 00 11 80 01 00 00 00 00 00
00000378 01 00 00 00 00 00 00 00 01 00 00 00 00 00 00 7E 12 02 00 00 00 00 00
00000390 01 00 00 00 00 00 00 00 01 00 00 00 00 00 00 80 16 02 00 00 00 00 00
```

Ordenação

A ordenação é feita utilizando apenas o acesso ao disco.

Merge Sort Externo

A complexidade $O(n \log(n))$.

Lembrando que cada bloco de termo tem tamanho de 24 bytes.

Alguns testes temporal utilizando a ordenação.

Bytes	Blocos	Tempo
25790232	1074593	117.857 seconds
2747616	114484	9.77569 seconds
1552728	64697	6.3397 seconds
680808	28367	1.99901 seconds
475200	19800	1.49599 seconds
395712	16488	0.920757 seconds
31848	1327	0.0907093 seconds

A ordenação é feita nos arquivos dos termos e no hash_table.tbl (alfabeticamente).

Testes

Tempo

A criação do índice invertido se mostrou muito lenta no arquivo `ri_2018_collection.dat` , levando 24 horas atingindo apenas 39014 páginas analisadas.


```

0.997680 seconds to read document 39002
173891 terms added.

Progress: 1% Position: 2412218691 Document ID: 39003
1.169221 seconds to read document 39003
175132 terms added.

Progress: 1% Position: 2412277349 Document ID: 39004
0.800832 seconds to read document 39004
173891 terms added.

Progress: 1% Position: 2412307714 Document ID: 39005
0.838978 seconds to read document 39005
175134 terms added.

Progress: 1% Position: 2412341368 Document ID: 39006
0.765874 seconds to read document 39006
173891 terms added.

Progress: 1% Position: 2412382696 Document ID: 39007
0.776896 seconds to read document 39007
175134 terms added.

Progress: 1% Position: 2412427357 Document ID: 39008
1.131615 seconds to read document 39008
173891 terms added.

Progress: 1% Position: 2412462672 Document ID: 39009
1.884037 seconds to read document 39009
175137 terms added.

Progress: 1% Position: 2412496451 Document ID: 39010
1.743768 seconds to read document 39010
173891 terms added.

Progress: 1% Position: 2412534511 Document ID: 39011
1.213798 seconds to read document 39011
175137 terms added.

Progress: 1% Position: 2412586920 Document ID: 39012
4.707040 seconds to read document 39012
173919 terms added.

Progress: 1% Position: 2412673693 Document ID: 39013
3.964296 seconds to read document 39013
175153 terms added.

Progress: 1% Position: 2412763821 Document ID: 39014
^CMakefile:19: recipe for target 'run' failed
make: *** [run] Interrupt

```

Os motivos são:

- latência ao acesso do HD externo.
- complexidade de busca no arquivo `hash_table.tb1` a cada termo. Se tiver **n** termos já adicionados no `hash_table.tb1` e a página provem **m** termos candidatos (que podem ser adicionados ou registrado a frequência), temos uma complexidade $O(nm)$ por página.

Um exemplo na tentativa de criar o arquivo invertido no `ri_2018_collection.dat`.

Após de 2.5GBs analisados, o `hash_table.tb1` se encontrava com 34MB e cada página levava em

média 3 segundos para ser analisada e aumentando logaritmicamente conforme o arquivo crescia. Tornando inviável coletar quase os 300GBs do arquivo.

O arquivo de testes fornecido `html_pages.txt` contendo 93MB gastam 16 minutos para criar o índice invertido.

```

yuri@Nala: ~/src/ufmg/2018_1_ri/tp2
yuri@Nala: ~/src/ufmg/2018_1_ri/tp2 118x62

1.120040 seconds to read document 1064
12 terms added.

Progress: 99% Position: 97096710 Document ID: 1065
2.321331 seconds to read document 1065
66 terms added.

Progress: 99% Position: 97232508 Document ID: 1066
1.102239 seconds to read document 1066
3 terms added.

Progress: 99% Position: 97244619 Document ID: 1067
0.563357 seconds to read document 1067
29 terms added.

Progress: 99% Position: 97282166 Document ID: 1068
3.557267 seconds to read document 1068
167 terms added.

Progress: 99% Position: 97392194 Document ID: 1069
0.122225 seconds to read document 1069
1 terms added.

Progress: 99% Position: 97401450 Document ID: 1070
0.312053 seconds to read document 1070
9 terms added.

Collection read takes:
974.439133 seconds to finish
There are 61026 terms.

*****
Init PASS 2

Sorting file 0
21 MB 22330 KB 22865952 bytes
elapsed time to sort 118.130331 seconds
Writing file 0
elapsed time to write 0.346437 seconds
Sorting file 1
2 MB 2683 KB 2747616 bytes
elapsed time to sort 11.611247 seconds
Writing file 1
elapsed time to write 0.190422 seconds
Sorting file 2
1 MB 1516 KB 1552728 bytes
elapsed time to sort 6.349112 seconds
Writing file 2
elapsed time to write 0.145037 seconds
Sorting file 3
0 MB 664 KB 680808 bytes
elapsed time to sort 2.385748 seconds
Writing file 3
elapsed time to write 0.199505 seconds
Sorting file 4
0 MB 464 KB 475200 bytes
elapsed time to sort 1.584242 seconds
Writing file 4
elapsed time to write 0.120390 seconds
Sorting file 5

```

Busca dos termos

Seguem *screenshots* com exemplos do programa `pass3`.

Busca simples de um termo

```

./pass3.out html_pages.txt
yuri@Nala: ~/src/ufmg/2018_1_ri/tp2 (master) ./pass3.out html_pages.txt
Search terms (Boolean mode)
type "/help" for more info.
> /help
Just type your terms separated with ' ' to search the available documents
Example: apoiam instagram
Commands:
/list
  print all terms available in hashtable file
/search <term> or /s <term>
  Search if term exist in hash table
/quit
  Close program
> /s yuri
Found term in position 58859
> /s yuria
Not found!
> apoiam
List of document's ids that found the term(s): <id, freq in doc>
id 3 freq 1
id 498 freq 1
id 696 freq 1
id 772 freq 1
id 891 freq 1
> xbox
List of document's ids that found the term(s): <id, freq in doc>
id 1 freq 2
id 22 freq 2
id 26 freq 2
id 55 freq 5
id 81 freq 2
id 82 freq 5
id 90 freq 5
id 112 freq 7
id 410 freq 9
id 438 freq 2
id 570 freq 7
id 617 freq 2
id 630 freq 2
id 650 freq 1
id 719 freq 1
id 746 freq 2
id 894 freq 9
id 956 freq 2
id 997 freq 9
>

# TP 2 - Recuperação de Informação
Este trabalho foi dividido em 3 passos.

A primeira parte 'pass1' consiste em criar os termos no arquivo 'terms.dump' mais as posições de mapeamento (posição/ponteiro) de cada índice em outro arquivo 'hash_table.tbl'. Os termos são guardados temporariamente em uma seção seguinte.

A segunda parte 'pass2' faz a ordenação e guarda em um único arquivo 'terms.index' (já ordenado e ordenado em ordem alfabética dos termos por posição).

A terceira parte 'pass3' é o programa que realiza a busca. Digitando '/help' lista algumas funções disponíveis. Alguns termos separados por espaços. O método utilizado para a busca é o de busca binária. Algumas funções disponíveis são:
- '/list': lista todos os termos junto com suas posições.
- '/s <term>' ou '/search <term>': busca o termo no índice.
- '/quit': fecha o programa.

## Execução

Compila em 3 binários 'pass1.out', 'pass2.out' e 'pass3.out'.

$ make

Apaga todos os arquivos criados para o índice.

$ make clean

Executa os 3 passos no arquivo 'html_pages.txt'.

$ make test

```

Busca com 3 termos simultâneos

```

./pass3.out html_pages.txt
yuri@Nala ~/src/ufmg/2018_1_ri/tp2 master • ./pass3.out html_pages.txt
Search terms (Boolean mode)
type "/help" for more info.
> windows
List of document's ids that found the term(s): <id, freq in doc>
id 1 freq 4
id 2 freq 6
id 17 freq 3
id 48 freq 2
id 55 freq 1
id 64 freq 10
id 81 freq 1
id 82 freq 1
id 90 freq 1
id 458 freq 1
id 535 freq 2
id 617 freq 2
id 623 freq 7
id 719 freq 1
id 894 freq 9
id 997 freq 4
id 1025 freq 1
id 1031 freq 2
> linux
List of document's ids that found the term(s): <id, freq in doc>
id 1 freq 1
id 64 freq 1
id 442 freq 1
id 847 freq 1
> xbox
List of document's ids that found the term(s): <id, freq in doc>
id 1 freq 2
id 22 freq 2
id 26 freq 2
id 55 freq 5
id 81 freq 2
id 82 freq 5
id 90 freq 5
id 112 freq 7
id 410 freq 9
id 438 freq 2
id 570 freq 7
id 617 freq 2
id 630 freq 2
id 650 freq 1
id 719 freq 1
id 746 freq 2
id 894 freq 9
id 956 freq 2
id 997 freq 9
> windows linux xbox
List of document's ids that found the term(s): <id>
1
>

```

```

# TP 2 - Recuperação de Informação
Este trabalho foi dividido em 3 passos.

- A primeira parte 'pass1' consiste em criar
dos termos no arquivo 'terms.dump' mais as in
arquivo em outro arquivo 'hash_table.tbl'. Ao
de mapeamento (posição/ponteiro) de cada i-e
'[nome do arquivo da coleção].index' e as lis
Os termos são guardados temporariamente em va
seção seguinte.

- A segunda parte 'pass2' faz a ordenação e a
em um único arquivo 'terms.index' (já ordena
é ordenado em ordem alfabética dos termos par

- A terceira parte 'pass3' é o programa que f
Digitando '/help' lista algumas funções dispo
os termos separados por espaços. O método uti
Algumas funções disponíveis são:
- '/list': lista todos os termos junto com
- '/s <termo>' ou '/search <termo>': ???
- '/quit': fecha o programa

## Execução

Compila em 3 binários 'pass1.out', 'pass2.out'

$ make

Apaga todos os arquivos criados para o índice

$ make clean

Executa os 3 passos no arquivo 'html_pages.tx

$ make test

Executa os 3 passos no arquivo 'ri_2018_colle

$ make run

Caso deseje voltar repetir os dois primeiros

```

Busca com 2 termos simultâneos

```

./pass3.out html_pages.txt
yuri@Nala ~/src/ufmg/2018_1_ri/tp2 master • ./pass3.out html_pages.txt
Search terms (Boolean mode)
type "/help" for more info.
> xbox
List of document's ids that found the term(s): <id, freq in doc>
id 1 freq 2
id 22 freq 2
id 26 freq 2
id 55 freq 5
id 81 freq 2
id 82 freq 5
id 90 freq 5
id 112 freq 7
id 410 freq 9
id 438 freq 2
id 570 freq 7
id 617 freq 2
id 630 freq 2
id 650 freq 1
id 719 freq 1
id 746 freq 2
id 894 freq 9
id 956 freq 2
id 997 freq 9
> windows
List of document's ids that found the term(s): <id, freq in doc>
id 1 freq 4
id 2 freq 6
id 17 freq 3
id 48 freq 2
id 55 freq 1
id 64 freq 10
id 81 freq 1
id 82 freq 1
id 90 freq 1
id 458 freq 1
id 535 freq 2
id 617 freq 2
id 623 freq 7
id 719 freq 1
id 894 freq 9
id 997 freq 4
id 1025 freq 1
id 1031 freq 2
> xbox windows
List of document's ids that found the term(s): <id>
1
55
81
82
90
617
719
894
997
>

```

```

A complexidade  $O(n \log(n))$ .
Lembrando que cada bloco de termo tem tama
Alguns testes temporal utilizando a ordena
Bytes | Blocos | Tempo |
| :----- | :----- | :----- |
| 25790232 | 1074593 | 117.857 seconds |
| 2747616 | 114484 | 9.77569 seconds |
| 1552728 | 64697 | 6.3397 seconds |
| 680808 | 28367 | 1.99901 seconds |
| 475200 | 19800 | 1.49599 seconds |
| 395712 | 16488 | 0.920757 seconds |
| 31848 | 1327 | 0.0907093 seconds |
A ordenação é feita nos arquivos dos termo
## Testes
A criação do índice invertido se mostrou m
levando 24 horas atingindo apenas 39014 pá
![[ri_2018_collection.png](img/ri_2018_coll
Os motivos são:
- latência ao acesso do HD externo.
- complexidade de busca no arquivo 'hash_t
já adicionados no 'hash table.tbl' e a pág
adicionados ou registrado a frequência),
Um exemplo na tentativa de criar o arquivo
Após de 2.5GBs analisados, o 'hash table.t
média 3 segundos para ser analisada e aume
inviável coletar quase os 300GBs
O arquivo de testes fornecido 'html pages.
criar o índice invertido.
![[html_pages.txt](img/html_pages.png)
### Busca dos termos

```