

Analyzing and Predicting LBUG's with a Hidden Markov Model

Thomas Stibor
t.stibor@gsi.de

High Performance Computing
GSI Helmholtz Centre for Heavy Ion Research
Darmstadt, Germany

Monday 22nd September, 2014

LAD'14 Workshop, Reims, France

Overview

- LBUG's in Lustre and implications.
- Process Lustre log data with log stash.
- Markov Model (MM).
- Hidden Markov Model (HMM).
- Example (from text processing) what HMM's can analyze.
- Frequency distributions of function calls.
- Visualized MM transition and HMM emission matrices.
- Sampling and predicting LBUG's in future time windows.

LBUG's and Implications

- Lustre is a large project with complex code base.
- Lustre is prone to **critical** software bugs called (LBUG's).
- LBUG is a software behavior that causes freeze of kernel thread and subsequent reboot.

```
void lbug_with_loc(struct libcfs_debug_msg_data *) __attribute__((noreturn));

#define LBUG() \
do { \
    LIBCFS_DEBUG_MSG_DATA_DECL(msgdata, D_EMERG, NULL); \
    lbug_with_loc(&msgdata); \
} while(0)

#define LIBCFS_DEBUG_MSG_DATA_DECL(dataname, mask, cdls) \
static struct libcfs_debug_msg_data dataname = { \
    .msg_subsys = DEBUG_SUBSYSTEM, \
    .msg_file = __FILE__, \
    .msg_fn = __FUNCTION__, \
    .msg_line = __LINE__, \
    .msg_cdls = (cdls), \
    dataname.msg_mask = (mask); \
}
```

"Note - LBUG freezes the thread to allow capture of the panic stack. A system reboot is needed to clear the thread."

LBUG's in Lustre Code Example

```
int mdt_getxattr(struct mdt_thread_info *info)
{
    struct ptlrpc_request *req = mdt_info_req(info);
    struct mdt_export_data *med = mdt_req2med(req);
    struct lu_ucred *uc = lu_ucred(info->mti_env);
    ...
    valid = info->mti_body->valid & (OBD_MD_FLXATTR | OBD_MD_FLXATTRLS);

    if (valid == OBD_MD_FLXATTR) {
        char *xattr_name = req_capsule_client_get(info->mti_pill,
                                                    &RMF_NAME);
        rc = mdt_getxattr_one(info, xattr_name, next, buf, med, uc);
    } else if (valid == OBD_MD_FLXATTRLS) {
        CDEBUG(D_INODE, "listxattr\n");

        rc = mo_xattr_list(info->mti_env, next, buf);
        if (rc < 0)
            CDEBUG(D_INFO, "listxattr failed: %d\n", rc);
    } else if (valid == OBD_MD_FLXATTRALL) {
        rc = mdt_getxattr_all(info, reqbody, repbody,
                              buf, next);
    } else
        LBUG();
}
```

Lustre Log Data Pre-Processing Steps

1. Fetching entire log data from archive tape.
2. Resulting in “giant” log data.

```
drwxr-sr-x 14 root staff 114 Mai 2 11:37 ..
-rw-r----- 1 root adm 2,5G Feb 2 2012 syslog-20120201
-rw-r----- 1 root adm 2,0G Feb 3 2012 syslog-20120202
-rw-r----- 1 root adm 2,2G Feb 4 2012 syslog-20120203
-rw-r----- 1 root adm 2,0G Feb 5 2012 syslog-20120204
-rw-r----- 1 root adm 1,9G Feb 6 2012 syslog-20120205
-rw-r----- 1 root adm 1,9G Feb 7 2012 syslog-20120206
(...)

du -sh 2012/02/
57G 02/
```

- Total amount of Lustre log data for 2012 is \approx 2.1 GByte.
- A true **Big Data** problem.

Processed Lustre Logs

Logstash: CSV exported data to be analyzed:

Example 1:

```
Mar 26 20:05:28, lxfs290, LustreError, filter.c, 2732, __filter_oa2dentry, testlust -
OST001f: filter_preprw_read on non-existent object: 10
Mar 26 20:05:28, lxfs290, LustreError, filter_io.c, 488, filter_preprw_read, ASSERTION
(PageLocked(lnb->page)) failed
Mar 26 20:05:28, lxfs290, LustreError, filter_io.c, 488, filter_preprw_read, LBUG
```

Example 2:

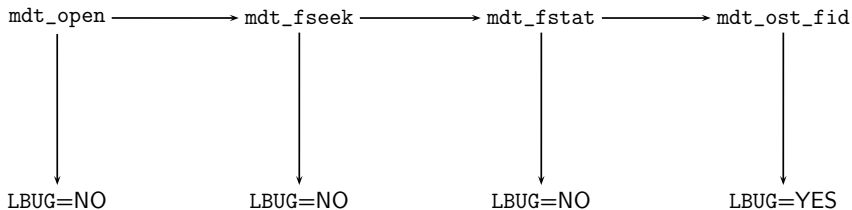
```
May 27 22:56:01, lxfs124, LustreError, events.c, 381, server_bulk_callback, "event
type 4, status -5, desc ffff8800c791c000"
May 28 00:15:50, lxmds11, LustreError, client.c, 178, ptlrpc_free_bulk, ASSERTION(
atomic_read(&(desc->bd_export)->exp_refcount) < 0x5a5a5a) failed
May 28 00:15:50, lxmds11, LustreError, service.c, 1426, ptlrpc_server_handle_request,
ASSERTION(atomic_read(&(export)->exp_refcount) < 0x5a5a5a) failed
May 28 00:15:50, lxmds11, LustreError, service.c, 1426, ptlrpc_server_handle_request,
LBUG
May 28 00:15:50, lxmds11, LustreError, client.c, 178, ptlrpc_free_bulk, LBUG
```

Note: Patch exists for Example 2

 Lustre / LU-919
Multiple wrong LBUGs checking cfs_atomic_t vars/fields with inaccurate poison value of 0x5a5a5a

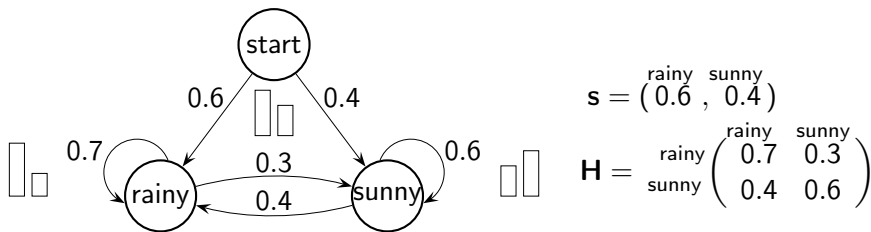
Can we predict LBUG's in Lustre?

Consider this simple approach for modeling Lustre *function calls* and corresponding LBUG occurrence:



This looks like the familiarized Hidden Markov model.

Markov Model Weather Example



- T : Length of observation sequence.
- N_Q : Number of states in the model.
- \mathbf{s} : Initial state distribution.
- \mathbf{H} : Transition matrix.
- $\{Q_1, Q_2, \dots, Q_T\}$: Set of time indexed random variables for states.
- $\mathfrak{M} = (\mathbf{s}, \mathbf{H})$: Markov model parameters.

Joint distribution for *sequence* of T observations

$$\Pr(Q_1, Q_2, \dots, Q_T) = \Pr(Q_1) \prod_{t=2}^T \Pr(Q_t | Q_1, \dots, Q_{t-1})$$

Joint distribution under *first-order* Markov assumption:

$$\Pr(Q_1, Q_2, \dots, Q_T) = \Pr(Q_1) \prod_{t=2}^T \Pr(Q_t | Q_{t-1})$$

Markov Model Weather Example (cont.)

Given that weather on day 1 ($t = 1$) is sunny.

- What is the probability for the next 3 days weather will be $\mathcal{O} = \text{"sunny} \rightarrow \text{rainy} \rightarrow \text{rainy"}$?

$$\begin{aligned}\Pr(\mathcal{O}|\mathfrak{M}) &= \Pr(Q_1 = \text{sunny}, Q_2 = \text{sunny}, Q_3 = \text{rainy}, Q_4 = \text{rainy}) \\ &= \Pr(Q_1 = \text{sunny}) \cdot \Pr(Q_2 = \text{sunny} \mid Q_1 = \text{sunny}) \\ &\quad \cdot \Pr(Q_3 = \text{rainy} \mid Q_2 = \text{sunny}) \\ &\quad \cdot \Pr(Q_4 = \text{rainy} \mid Q_3 = \text{rainy}) \\ &= \mathbf{s}_2 \cdot \mathbf{H}_{22} \cdot \mathbf{H}_{21} \cdot \mathbf{H}_{11} \\ &= 0.4 \cdot 0.6 \cdot 0.4 \cdot 0.7 = 0.0672\end{aligned}$$

Note: Entries in matrix \mathbf{H} can be interpreted as follows:

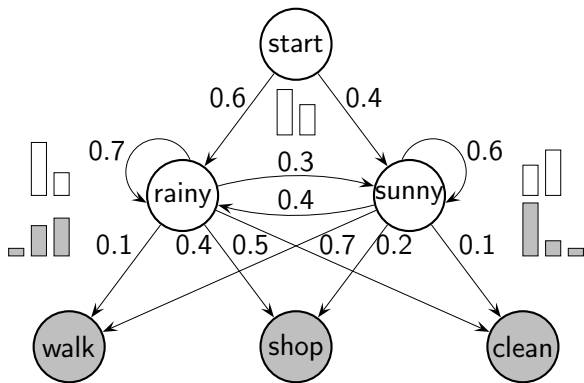
$$\mathbf{H}_{ij} = \Pr(Q_{t+1} = j \mid Q_t = i)$$

where for the sake of simplicity states are from set $\{1, 2, \dots, N_Q\}$

Hidden Markov models

Suppose we are locked in room without windows, and somebody is telling us the following observations and ask us to tell him what weather is outside:

$\mathcal{O} = \text{walk} \rightarrow \text{clean} \rightarrow \text{shop} \rightarrow \dots \rightarrow \text{shop}$



$$\mathbf{s} = \begin{matrix} \text{rainy} & \text{sunny} \\ (0.6, & 0.4) \end{matrix}$$

$$\mathbf{H} = \begin{matrix} & \text{rainy} & \text{sunny} \\ \text{rainy} & \begin{pmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{pmatrix} \\ \text{sunny} & \end{matrix}$$

$$\mathbf{E} = \begin{matrix} & \text{walk} & \text{shop} & \text{clean} \\ \text{rainy} & \begin{pmatrix} 0.1 & 0.4 & 0.5 \\ 0.7 & 0.2 & 0.1 \end{pmatrix} \\ \text{sunny} & \end{matrix}$$

Hidden Markov models (cont.)

Problem 1: Calculate probability of observation sequence \mathcal{O} , given model \mathfrak{M} , that is $\Pr(\mathcal{O} | \mathfrak{M}) = ?$

Problem 2: Given HMM and observation sequence \mathcal{O} , find most likely hidden state sequence.

Problem 3: How do we estimate model parameters $\mathfrak{M} = (\mathbf{s}, \mathbf{H}, \mathbf{E})$ to maximize $\Pr(\mathcal{O} | \mathfrak{M})$. Loosely speaking, how do we estimate \mathfrak{M} that “best” fits our data \mathcal{O} .

For further details see paper:

- *Lawrence R. Rabiner*. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE, 1989, pages 257-286.

Analyze Novel Flatland with a HMM

Example from Sec. 1 *Of the Nature of Flatland*: by Edwin A. Abbott

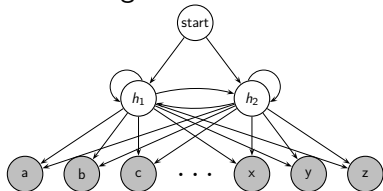
I call our world Flatland, not because we call it so, but to make its nature clearer to you, my happy readers, who are privileged to live in Space.

Imagine a vast sheet of paper on which straight Lines, Triangles, Squares, Pentagons, Hexagons, and other figures, ...

Process and convert text into:

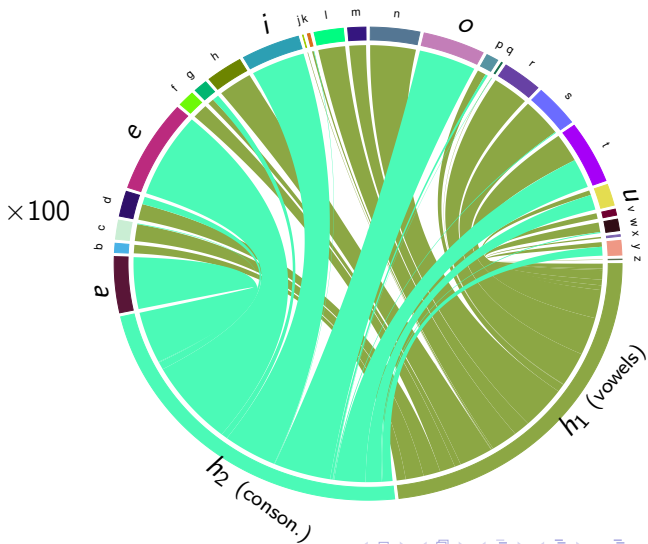
i c a l l o u r w o r l d f l a t l a n d n o t b e c a u s e w e c a l l i t
s o b u t t o m a k e i t s n a t u r e c l e a r e r t o y o u m y h a p p y
r e a d e r s w h o a r e p r i v i l e g e d t o l i v e i n s p a c e i m a
g i n e a v a s t s h e e t o f p a p e r o n w h i c h s t r a i g h t l i n
e s t r i a n g l e s s q u a r e s p e n t a g o n s h e x a g o n s a n d o
t h e r f i g u r e s

Train HMM of following form:



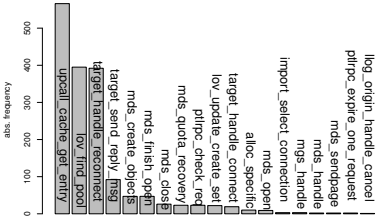
Analyze Novel Flatland with a HMM (cont.)

letter	h_1	h_2
a	0.00	16.51
b	2.70	0.00
c	5.37	0.24
d	5.06	2.39
e	0.00	27.10
f	4.59	0.00
g	1.84	1.92
h	10.10	0.00
i	0.00	16.14
j	0.16	0.00
k	0.50	0.16
l	8.02	0.00
m	4.97	0.00
n	13.58	0.00
o	0.00	17.03
p	2.55	0.77
q	0.13	0.13
r	10.90	0.00
s	12.06	0.30
t	9.03	9.29
u	1.49	4.80
v	1.78	0.00
w	3.04	0.34
x	0.46	0.00
y	1.46	2.82
z	0.11	0.00

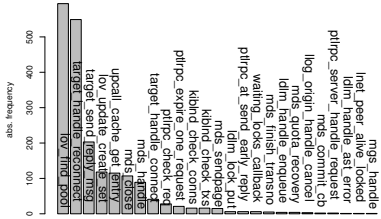


Frequency Distribution of Function Calls (1-Gram Seq.)

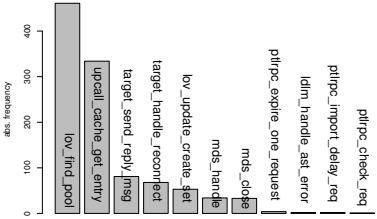
lxmnds11 function call distribution 01–2013



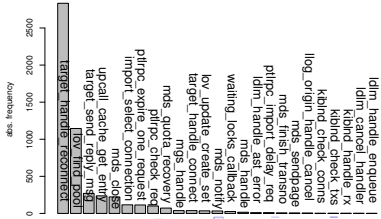
lxmnds11 function call distribution 02–2013



lxmnds11 function call distribution 03–2013

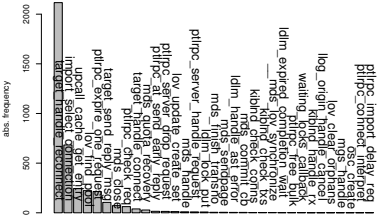


lxmnds11 function call distribution 04–2013

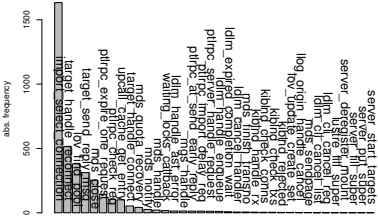


Frequency Distribution of Func. Calls (1-Gram Seq.) (cont.)

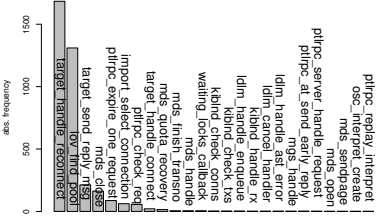
lxmnds11 function call distribution 05-2013



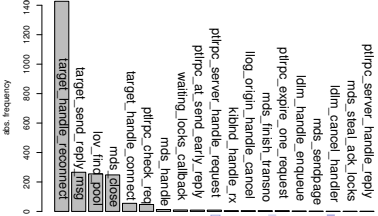
lxmnds11 function call distribution 06-2013



lxmnds11 function call distribution 07-2013

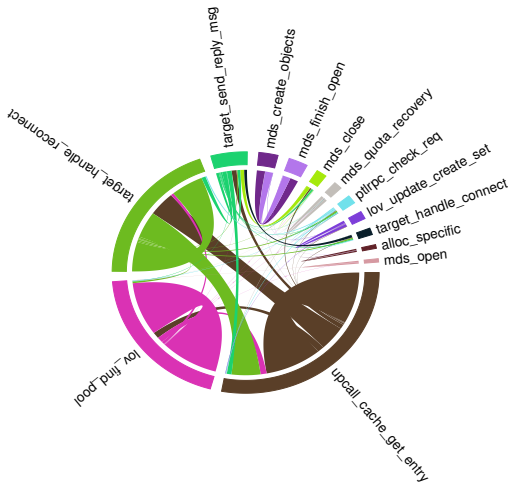


lxmnds11 function call distribution 08-2013



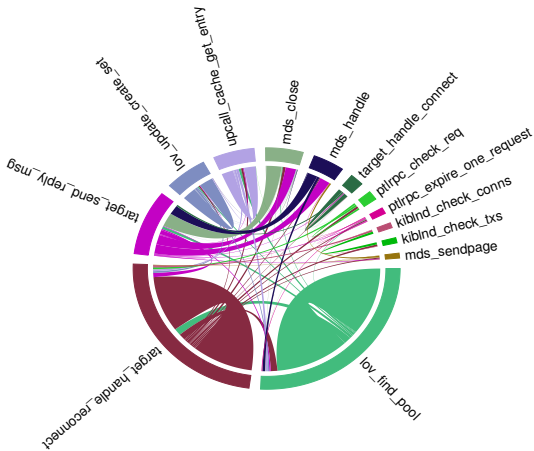
Visualize Markov Model (2-Gram Sequence) of Fnc. Calls

ixmds11-01-2013.seq



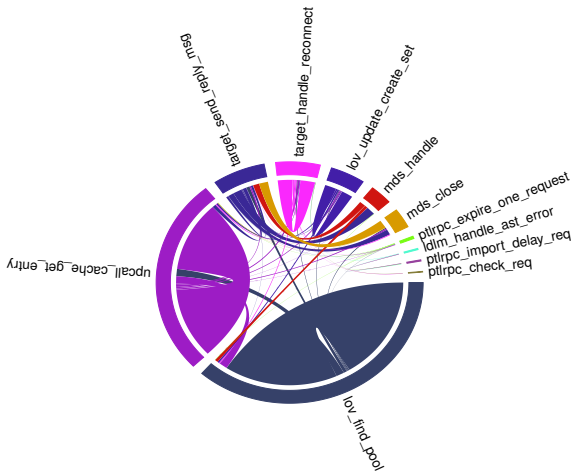
Visualize Markov M. (2-Gram Seq.) of Fnc. Calls (cont.)

lxmds11-02-2013.seq



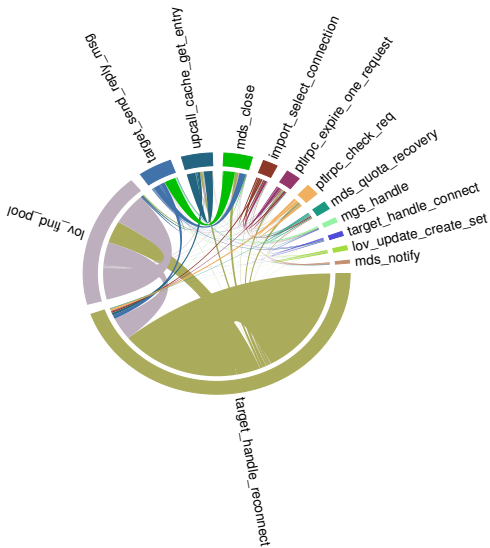
Visualize Markov M. (2-Gram Seq.) of Fnc. Calls (cont.)

lxmds11-03-2013.seq



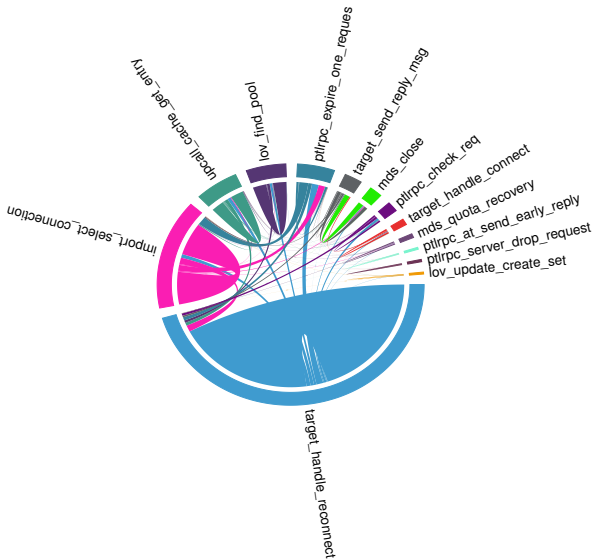
Visualize Markov M. (2-Gram Seq.) of Fnc. Calls (cont.)

lxmds11-04-2013.seq



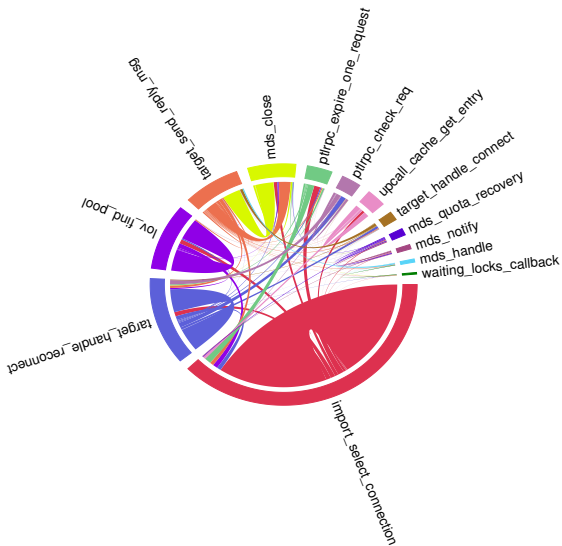
Visualize Markov M. (2-Gram Seq.) of Fnc. Calls (cont.)

lxmds11-05-2013.seq



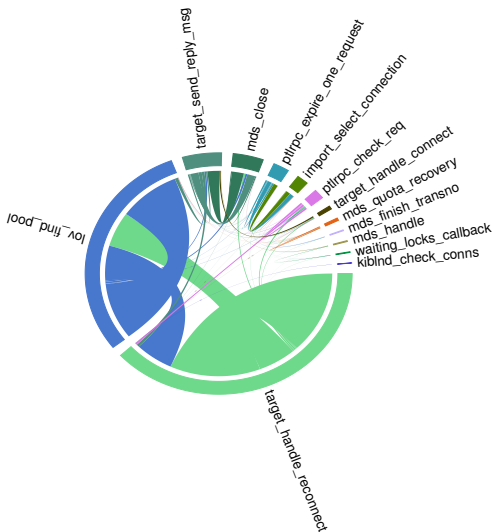
Visualize Markov M. (2-Gram Seq.) of Fnc. Calls (cont.)

lxmds11-06-2013.seq



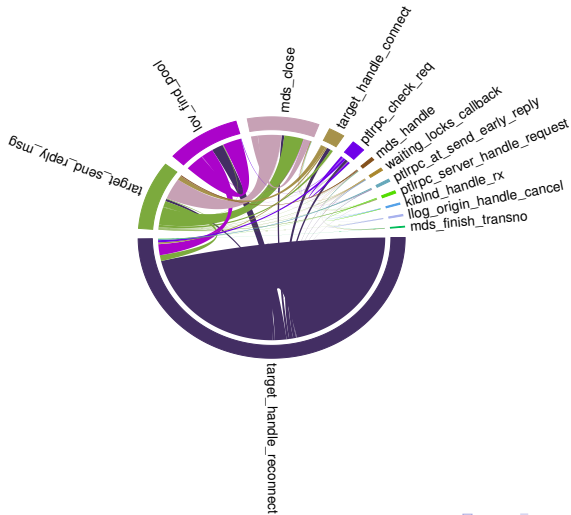
Visualize Markov M. (2-Gram Seq.) of Fnc. Calls (cont.)

lxmds11-07-2013.seq



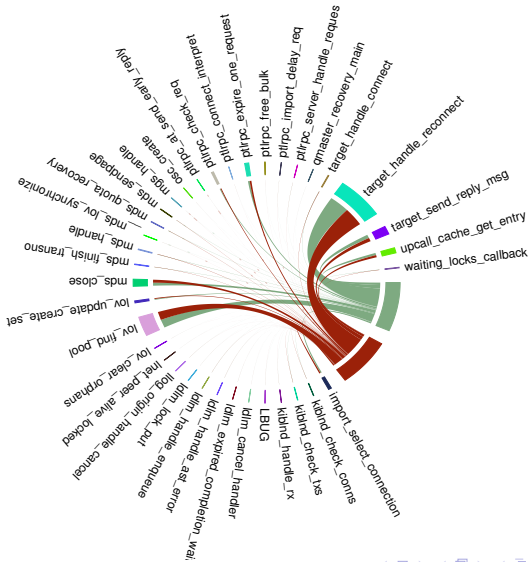
Visualize Markov M. (2-Gram Seq.) of Fnc. Calls (cont.)

lxmds11-08-2013.seq



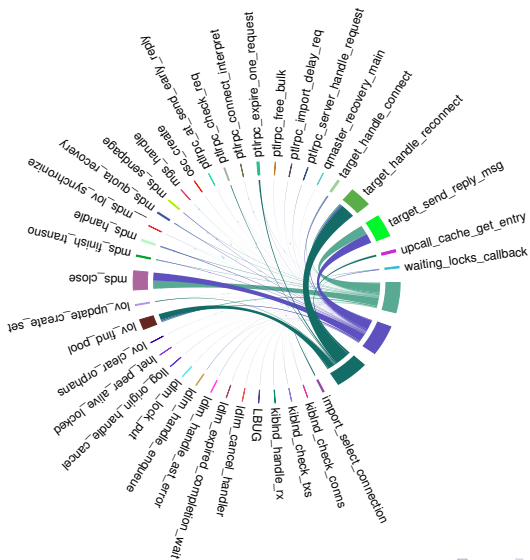
Visualize HMM of Functions Calls

2 Hidden States



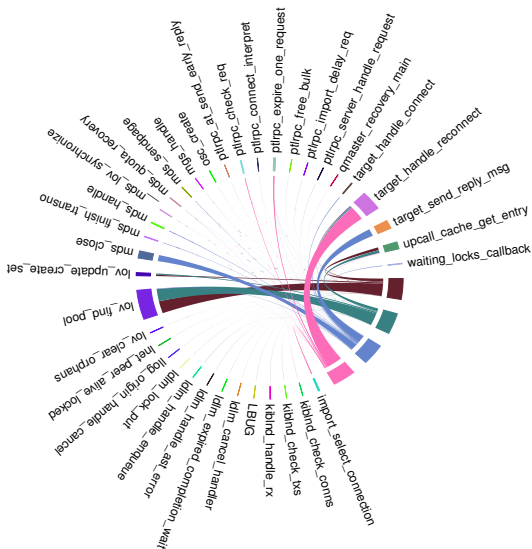
Visualize HMM of Functions Calls (cont.)

3 Hidden States



Visualize HMM of Functions Calls (cont.)

4 Hidden States

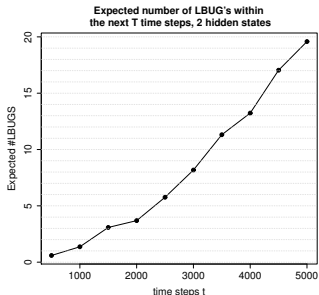


LBUG Prediction with HMM

- Train HMM on Lustre logs (function calls, LBUGs)
- Sample function call sequences from HMM.

Example: Sample from trained HMM (2 hidden states, 36 emitting states (function calls))

```
ldlm_handle_enqueue , lov_find_pool , target_handle_connect ,  
target_handle_reconnect , target_handle_reconnect , target_handle_reconnect ,  
lov_find_pool , target_send_reply_msg , mds_close ,  
lov_clear_orphans , .... , target_handle_reconnect , target_handle_reconnect ,  
target_handle_reconnect , LBUG
```



Summary & Outlook

Analyzing Lustre log files with (Hidden) Markov Models for:

- visualizing and analyzing problems,
- recover latent structures and relations,
- predicting future problems (LBUG's), by sampling from the model.

HMM implementation (in C) and R scripts (for generating chord diagrams) will be available soon at <https://bitbucket.org/tstibor>

Many thanks to [Matteo Dessalvi](#) for providing Lustre log data and great discussions on machine learning.

Questions ?