WCCL - Błąd #3845
Some data-driven tests fail on system with en_US.UTF-8 locale

12 Mar 2012 11:49 - Adam Radziszewski

Status:	Rozwiązany	**Start date:**	12 Mar 2012
Priority:	Normalny	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

The likely reason is different collation assumed, although this is still odd as the problem also appears with ASCI-only characters. The tests fail with different order of strset, e.g.:

```
Expected: ["ąca", "ący", "ca"]
Actual:   ["ca", "ący", "ąca"]
```

Another test case may be the constant ["psi", "pies"], which appears as ["pies", "psi"] under Ubuntu 10.04, LANG=pl_PL.utf8, LANGUAGE=en_GB:en, while being output as ["psi", "pies"] under Ubuntu 11.10, en_US.UTF-8. The results are obtained with:

```
wccl-run examples/in-xces.xml '["psi", "pies"]'
```

It is not obvious whether the problem is locale-related, although non-determinism is clearly visible.

History

#1 - 12 Mar 2012 11:52 - Adam Radziszewski

This non-deterministic behaviour may also result in unpredictable output of a system that uses string representations of multi-value string sets obtained from WCCL.

#2 - 12 Mar 2012 11:56 - Adam Radziszewski

It seems like an issue with (lack of) sorting of the output set when generating str repr:

```
eliasz@ubu11-VirtualBox:~/wccl/bin$ wccl-parser
Enter any operator expression: ["pies", "psi"]
[ 0] Parsed expression: ["pies", "psi"]
Enter any operator expression: ["psi", "pies"]
[ 0] Parsed expression: ["psi", "pies"]
```

#3 - 12 Mar 2012 12:14 - Adam Radziszewski

Ok, the underlying type is unordered_set, while string representation routines (all of them!) use plain const_iterator. Those routines need serious refactoring, anyway (values/strset.cpp).

#4 - 12 Mar 2012 13:32 - Bartosz Broda

It would be good to document this behavior in the code... At the very least a url in the code to this issue should be given.

#5 - 20 Apr 2012 14:48 - Adam Radziszewski

- Status changed from Nowy to Rozwiązany

Solved by changing the underlying unordered_set to std::set, which btw seems to boost performance.

Test case (run 10 times):

```
/usr/bin/time --format="%e" wccl-run -t nkjp strset.ccl  ~/NKJP-10/folds/train01.xml > s.txt
```

```
// strset.ccl -- gathering all the base forms from range between focus position (0) and sent end
@ "str" (
  if(
    rlook(0, end, $I,
      not(setvar($s:B, union($s:B, base[$I])))
    ),
    $s:B,
    $s:B
  )
)
```

unordered_set: 116.454 s average (std dev: 3.094 s)
std::set: 94.701 average (std dev: 1.206 s)

#6 - 20 Apr 2012 20:20 - Bartosz Broda

Adam Radziszewski wrote:

> *unordered_set: 116.454 s average (std dev: 3.094 s)*
> *std::set: 94.701 average (std dev: 1.206 s)*

This results are flabbergasting! Care to try simple std::vector?