A New Standard ML Prettyprinter Library

An Experience Report

David MacQueen

Department of Computer Science (Emeritus)
University of Chicago

September 8, 2023

Why? YAPPL?

- A more *Natural*, *Simple*, *Predictable* prettyprinting model (than PP/Format)
- Expressive enough for common usage – modest goals
- Why not be satisfied with an SML port of Wadler-Leijen?
  Not natural and simple enough!

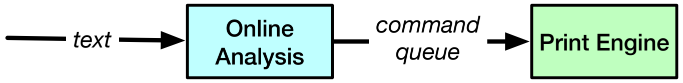
This talk: An advertisement, not a tutorial

A Little History

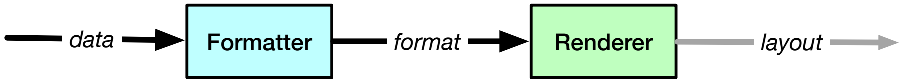
1. Oppen, 1980: online, limited length queue of “tokens”
2. PPML, 1986: *boxes*, conditional line breaks, alignments
3. Hughes-Wadler 1995-97:
   - *document* intermediate structure
   - *algebraic* development
   - lazy evaluation *not* necessary

Oppen and Hughes-Wadler models

Oppen model



H-W model



Goals and Non-Goals

- Motivated by need to print compiler representations (ASTs, types, IRs) for compiler debugging and compiler responses, including error messages
- Not trying to compete with text markup languages (markdown, asciidoc, LaTeX)

Main Ideas

- *blocks* (nested compound formats) [analagous to “boxes” in PPML, Knuth’s TeX].
- *breaks* (hard and soft line breaks, spaces)
- *alignments* (Compact, Horizontal, Vertical, Packed)
- simple, conservative block *measure* (“flat” measure)
- 1.5-dimensional layout strategy (line length constraint, 1 block look ahead)

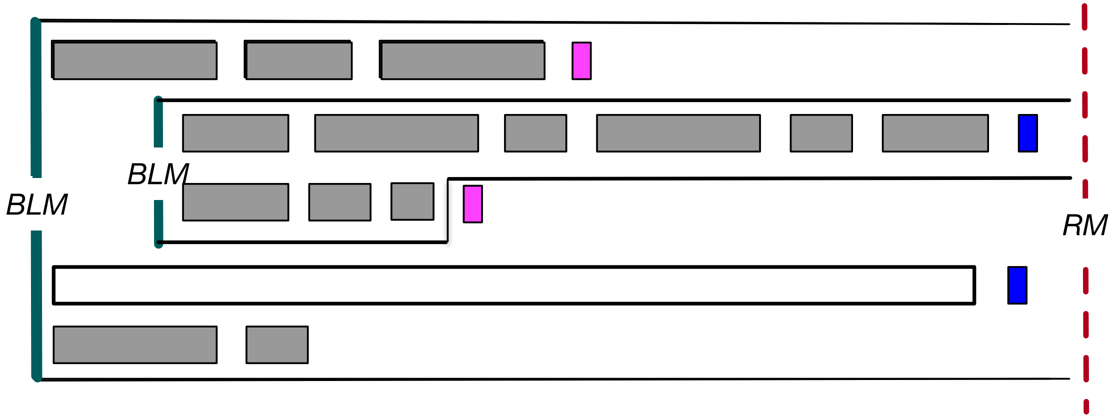
The format Type

```
datatype break = Null | Hard | Soft of int | Space of int

datatype alignment = C | H | V | P

datatype format  (* slightly simplified *)
  = EMPTY
  | TEXT of string
  | BLOCK of element list  (* element = break + format *)
  | ABLOCK of alignment * format list
  | INDENT of int * format
  | FLAT of format
  | ALT of format * format
```

Example: Nested Block Structure



Conditional Rendering

At a soft line break:

`BLOCK[...,Soft n,format,...]` or `ABLOCK(P, [..., format, ...]`

Will the following format fit on the current line?

No? Then trigger the soft line break.

`alt(format1, format2)`:

Render `format1` *if it "fits", otherwise render* `format2`.

Format Measurement (for estimating “fit”)

- *Flat measure*
  (as though rendered on an unbounded line, suppressing all line breaks)
- Measuring in characters, assuming a monospace font

Indented formats

- `INDENT` (`indent n`) is a format modifier applying to an entire format
- Indentation is activated only if immediately preceeded by a line break

Example: Wadler's trees

```
datatype tree = N of string * tree list

(* fmtTree : tree -> format *)
fun fmtTree (N (s, trees)) =
    cblock [text s, fmtTrees trees]

(* fmtTrees : tree list -> format *)
and fmtTrees nil = empty
  | fmtTrees trees =
      brackets (vsequence comma (map fmtTree trees))
```

Example: Wadler's trees

```
val tree1 =
    N ("aaa",
       [N ("bbbbb",
           [N ("ccc", nil),
            N ("dd", nil)]),
        N ("eee", nil),
        N ("ffff",
           [N ("gg", nil),
            N ("hhh", nil),
            N ("ii", nil)])]);
```

```
- printFormatNL (fmtTree tree1);
aaa[bbbbb[ccc,
          dd],
    eee,
    ffff[gg,
         hhh,
         ii]]
```

Experience

- Re-implemented 11 prettyprinters within the SML/NJ system, translating them from older versions that used the PP library (plus assorted error messages)
- Need further testing and "tuning"
- Need feedback from external users

Enhancements and potential new features

Done:

- *styled text* (emphasis, color), *e.g.*, rendering to ANSI terminals, HTML
- *fancy text* (UTF8)

Potential:

- Refined line break control: *e.g.*, ribbon percentage
- Tabs: basic or “scoped” (Westrick)

Conclusions

- Simple model (nested blocks, alignment, conditional line breaks, indented blocks)
- Simple implementation (purely functional, strict)
- Efficient enough (linear?)
  Only significant algorithmic idea is memoization of block measures
  (What would optimality mean?)
- The library should be easy to port to other languages (*e.g.*, OCaml, Haskell, Python, *etc.*).

Availability

Source code and documentation available at:

```
www.github.com/smlnj/prettyprint
```

Questions?