PageLayout InfluenceOfPageSizesOnAlgorithm

Influence of different page sizes on the pagination algorithm to be developed

XSL-FO allows arbitrary sequences of different page specifications. The consequence is that you may have different available IPDs for any page.

Many of the papers we found on pagination don't seem to deal with this fact. They focus mainly on formatting books where each page has the same available IPD. That allows for prefabrication of the vertical box lists and only the page breaking decisions have to be made and the individual boxes distributed over the pages.

An example found in [1], chapter 2.1: "As to the text stream, we assume that the line breaking task has already been completed. Afterwards, each paragraph consists of a sequence of line boxes and vertical glue boxes."

If we consider the possibility of different available IPDs over pages/regions, we can see that for each possible page break a different set of vertical boxes is generated on the subsequent page. This gets worse if a total-fit algorithm is chosen and a whole page-sequence is considered. The number of combinations grows exponentially.

If the available IPD remains equal between pages (which is true for most documents) look-ahead is possible and bearable from the performance POV. If a look-ahead algorithm is chosen, it must be capable of switching off look-ahead if subsequent pages have different available IPD, thus reducing it to a simple best-fit or first-fit algorithm. So if the available IPD remains equal the generated vertical boxes can be reused on a subsequent pages, they only have to be discarded if the IPD changes.

It remains to be determined if side-floats have an impact here. Probably not, since they mostly affect only the present page and there only the line-breaking.

Example of such a document: http://people.apache.org/~jeremias/fop/alternating-page-sizes.fo, http://people.apache.org/~jeremias/fop/alternating-page-sizes.pdf

---

[1] Brüggemann-Klein, Klein, Wohlfeil: On the Pagination of Complex Documents, http://www.pi6.fernuni-hagen.de/publ/tr234.pdf

---

Details to consider when implementing "changing available IPD" in FOP Trunk

Here are some points we should look into when we're planning on implementing support for "changing available IPD" in FOP Trunk:

- The algorithm for page breaking has to be changed from total-fit to best-fit, either be modifying the existing breaking algorithm in the PageBreakingAlgorithm subclass to behave like best-fit or by implementing a new algorithm based on Donald Knuths description.
- The LMs have to be restartable. Line breaks become invalid when the available IPD changes and they have to be recalculated, so the LMs have to be able to setup at an earlier point and restart element list generation from there. We probably need something similar to what we had before introducing the Knuth approach (see resetPosition() methods). Looks like we removed that prematurely.
    - The table and list LMs currently don't support even resuming element list generation after a hard break as the other LMs do.
- A suitable approach has to be found so that not too many precalculated elements have to be discarded when the available IPD changes.
- We have to investigate how the change of the page breaking algorithm has impact on features such as footnotes, floats and page-position="last".
- Column balancing for multi-columns documents relies on the total-fit algorithm. We will have to find a different approach here.
- The table LM have to be able deal with changing IPD. The columns setup, for example, may change from page to page for a table that spans multiple pages, especially if auto table layout is active.