LuaTEX as TEX successor

David Kastrup[1]

May 2, 2008

[1] dak@gnu.org

Case summary

Case summary

Friends, Romannumerals, Countrymen, lend me your ears
I come to defend TeX, not to praise it

Case summary

Friends, Romannumerals, Countrymen, lend me your ears
I come to defend TEX, not to praise it

- TEX had a terribly hard childhood

Case summary

Friends, Romannumerals, Countrymen, lend me your ears
I come to defend TEX, not to praise it

- TEX had a terribly hard childhood
- Single father

Case summary

Friends, Romannumerals, Countrymen, lend me your ears
I come to defend TEX, not to praise it

- TEX had a terribly hard childhood
- Single father
- Artificial mother tongue (stripped down Pascal)

Case summary

Friends, Romannumerals, Countrymen, lend me your ears
I come to defend TEX, not to praise it

- TEX had a terribly hard childhood
- Single father
- Artificial mother tongue (stripped down Pascal)
- Never got to know dynamic allocation

Case summary

Friends, Romannumerals, Countrymen, lend me your ears
I come to defend TEX, not to praise it

- TEX had a terribly hard childhood
- Single father
- Artificial mother tongue (stripped down Pascal)
- Never got to know dynamic allocation
- No family or friends of its own age

Case summary

Friends, Romannumerals, Countrymen, lend me your ears
I come to defend TeX, not to praise it

- TeX had a terribly hard childhood
- Single father
- Artificial mother tongue (stripped down Pascal)
- Never got to know dynamic allocation
- No family or friends of its own age
- There was no computer science or laws for it in its youth

Case summary

Friends, Romannumerals, Countrymen, lend me your ears
I come to defend TEX, not to praise it

- TEX had a terribly hard childhood
- Single father
- Artificial mother tongue (stripped down Pascal)
- Never got to know dynamic allocation
- No family or friends of its own age
- There was no computer science or laws for it in its youth
- It broke the path to tell people about computer science, and now is being judged by the laws that it helped create

Managable problems

Managable problems

- Simplest measures such as `\boxstretch`, `\boxfilstretch`, `\boxshrink` etc are not available.

Managable problems

- Simplest measures such as `\boxstretch`, `\boxfilstretch`, `\boxshrink` etc are not available.
- Boxes can't reliably be deconstructed (`\special`, single characters etc. can't be removed, boxes can only be taken apart from the end)

Managable problems

- Simplest measures such as `\boxstretch`, `\boxfilstretch`, `\boxshrink` etc are not available.
- Boxes can't reliably be deconstructed (`\special`, single characters etc. can't be removed, boxes can only be taken apart from the end)
- Variables that TeX employs for decisions are partly unavailable (in some cases because of system-dependent rounding)

Managable problems

- Simplest measures such as `\boxstretch`, `\boxfilstretch`, `\boxshrink` etc are not available.
- Boxes can't reliably be deconstructed (`\special`, single characters etc. can't be removed, boxes can only be taken apart from the end)
- Variables that TEX employs for decisions are partly unavailable (in some cases because of system-dependent rounding)
- Peculiarities like the loss of the first line's baseline (for `\vtop`) by whatsits, `\splittopskip0pt` and other.

Managable problems

- Simplest measures such as `\boxstretch`, `\boxfilstretch`, `\boxshrink` etc are not available.
- Boxes can't reliably be deconstructed (`\special`, single characters etc. can't be removed, boxes can only be taken apart from the end)
- Variables that TEX employs for decisions are partly unavailable (in some cases because of system-dependent rounding)
- Peculiarities like the loss of the first line's baseline (for `\vtop`) by whatsits, `\splittopskip0pt` and other.

LuaTEX: Partly done, certainly doable.

Problems of the macro language

Problems of the macro language

- Only global register pools indexed by number are available. There are no lexically local variables, the grouping structure does not match the macro structure.

Problems of the macro language

- Only global register pools indexed by number are available. There are no lexically local variables, the grouping structure does not match the macro structure.
- macro arguments get `\catcode` too soon, complex patterns are not easily parseable. Maybe `\lazy\def` would help?

Problems of the macro language

- Only global register pools indexed by number are available. There are no lexically local variables, the grouping structure does not match the macro structure.
- macro arguments get `\catcode` too soon, complex patterns are not easily parseable. Maybe `\lazy\def` would help?
- Implementing regular input languages is hard.

Problems of the macro language

- Only global register pools indexed by number are available. There are no lexically local variables, the grouping structure does not match the macro structure.
- macro arguments get `\catcode` too soon, complex patterns are not easily parseable. Maybe `\lazy\def` would help?
- Implementing regular input languages is hard.

LuaTeX: Some issues appear addressable, but the solutions do not interact closely with TeX regarding the data types and structures.

Interoperation problems

TEX

Interoperation problems

TeX

- only knows its own font formats, metrics and ligatures.

Interoperation problems

TeX

- only knows its own font formats, metrics and ligatures.
- does not talk to graphic programs

Interoperation problems

TeX

- only knows its own font formats, metrics and ligatures.
- does not talk to graphic programs
- can't trigger reformatting of external material.

Interoperation problems

TEX

- only knows its own font formats, metrics and ligatures.
- does not talk to graphic programs
- can't trigger reformatting of external material.

LuaTEX: Has the basics for understanding OpenType, talks with mplib, can call external programs.

Algorithmic problems

Algorithmic problems

- TEX is either perfect, or deficient: paragraphs are optimized globally, but the vertical breaks are “local best fit” without feedback to horizontal breaks or future pages.

Algorithmic problems

- TEX is either perfect, or deficient: paragraphs are optimized globally, but the vertical breaks are “local best fit” without feedback to horizontal breaks or future pages.
- TEX has no sane concept for asynchronous user code. `\output` is shielded with the expedient of additional grouping and has no multithreading concept.

Algorithmic problems

- TeX is either perfect, or deficient: paragraphs are optimized globally, but the vertical breaks are "local best fit" without feedback to horizontal breaks or future pages.
- TeX has no sane concept for asynchronous user code. `\output` is shielded with the expedient of additional grouping and has no multithreading concept.
- TeX has no possibilities for making use of side-effect free user-defined code. Consequently, user-defined code can't be used in several speculative contexts.

Algorithmic problems

- TEX is either perfect, or deficient: paragraphs are optimized globally, but the vertical breaks are "local best fit" without feedback to horizontal breaks or future pages.
- TEX has no sane concept for asynchronous user code. `\output` is shielded with the expedient of additional grouping and has no multithreading concept.
- TEX has no possibilities for making use of side-effect free user-defined code. Consequently, user-defined code can't be used in several speculative contexts.

LuaTEX: Does not touch TEX's algorithms here, only taps into them. Implications of `\output`, CoCo and coroutines interesting. Does not mesh with TEX.

Task at hand

If your ultimate goal is to produce a set of files in a different format that can be produced by GhostScript, take a look at the `tightpage` option of the preview package. This will embed the page dimensions into the PostScript code, obliterating the need to use the `-E -i` options to Dvips. You can then produce all image files with a single run of GhostScript from a single PostScript file for all images at once. The `tightpage` option requires setting the `dvips` option as well.

Various options exist that will pass TEX dimensions and other information about the respective shipped out material (including descender size) into the log file, where external applications might make use of it.

The possibility for generating a whole set of graphics with a single run of LATEX, Dvips, and GhostScript increases both speed and robustness of applications. It is to be hoped that applications like LATEX2HTML will be able to make use of this package in future.

Current line number implementations

Implementation with `lineno.sty`:

Current line number implementations

Implementation with `lineno.sty`:

1. Replaces all interline penalties with forced page breaks.

Current line number implementations

Implementation with `lineno.sty`:

1. Replaces all interline penalties with forced page breaks.
2. This triggers a special output routine placed before the principal output routine.

Current line number implementations

Implementation with `lineno.sty`:

1. Replaces all interline penalties with forced page breaks.
2. This triggers a special output routine placed before the principal output routine.
3. This special routine places the line numbers and reinserts the correct penalties.

Current line number implementations

Implementation with `lineno.sty`:

1. Replaces all interline penalties with forced page breaks.
2. This triggers a special output routine placed before the principal output routine.
3. This special routine places the line numbers and reinserts the correct penalties.
4. The normal Output routine is called.

Current line number implementations

Implementation with `lineno.sty`:

1. Replaces all interline penalties with forced page breaks.
2. This triggers a special output routine placed before the principal output routine.
3. This special routine places the line numbers and reinserts the correct penalties.
4. The normal Output routine is called.
5. A label-like multipass mechanism resets line numbers at the start of the page.

What would be saner for line numbering?

What would be saner for line numbering?

1. For migrating boxes into the main vertical list, a special "context" is defined that assembles a parallel column of 'unfinished' line numbers.

What would be saner for line numbering?

1. For migrating boxes into the main vertical list, a special “context” is defined that assembles a parallel column of ‘unfinished’ line numbers.
2. The unfinished objects take up constant dimensions and will be translated into glyphs either in the context of the output routine or at shipout time, since then the page start is known.

What would be saner for line numbering?

1. For migrating boxes into the main vertical list, a special “context” is defined that assembles a parallel column of ‘unfinished’ line numbers.
2. The unfinished objects take up constant dimensions and will be translated into glyphs either in the context of the output routine or at shipout time, since then the page start is known.
3. Consequently, a multipass algorithm is not necessary.

What would be saner for line numbering?

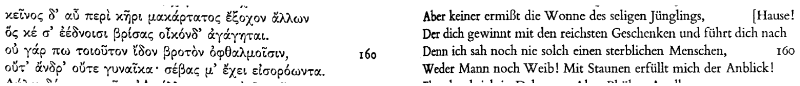
1. For migrating boxes into the main vertical list, a special "context" is defined that assembles a parallel column of 'unfinished' line numbers.
2. The unfinished objects take up constant dimensions and will be translated into glyphs either in the context of the output routine or at shipout time, since then the page start is known.
3. Consequently, a multipass algorithm is not necessary.
4. In the same context `\label`-commands referencing line numbers are expanded.

What would be saner for line numbering?

1. For migrating boxes into the main vertical list, a special “context” is defined that assembles a parallel column of ‘unfinished’ line numbers.
2. The unfinished objects take up constant dimensions and will be translated into glyphs either in the context of the output routine or at shipout time, since then the page start is known.
3. Consequently, a multipass algorithm is not necessary.
4. In the same context `\label`-commands referencing line numbers are expanded.

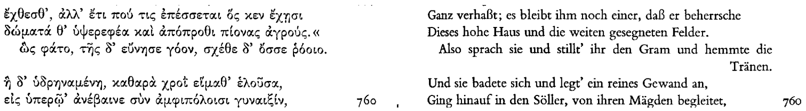
LuaTEX: Node list deconstruction/manipulation is most likely the easiest way. “Migration” is not a concept in LuaTEX, but could be interesting.

Synchronized texts. . .



κεῖνος δ' αὖ περὶ κῆρι μακάρτατος ἔξοχον ἄλλων
ὅς κέ σ' ἐέδνοισι βρίσας οἶκόνδ' ἀγάγηται.
οὐ γάρ πω τοιοῦτον ἴδον βροτὸν ὀφθαλμοῖσιν,
οὔτ' ἄνδρ' οὔτε γυναῖκα· σέβας μ' ἔχει εἰσορόωντα.

Aber keiner ermißt die Wonne des seligen Jünglings, [Hause!
Der dich gewinnt mit den reichsten Geschenken und führt dich nach
Denn ich sah noch nie solch einen sterblichen Menschen,
Weder Mann noch Weib! Mit Staunen erfüllt mich der Anblick!



ἔχθεσθ', ἀλλ' ἔτι πού τις ἐπέσσεται ὅς κεν ἔχῃσι
δώματά θ' ὑψερεφέα καὶ ἀπόπροθι πίονας ἀγρούς.«
ὣς φάτο, τῆς δ' εὔνησε γόον, σχέθε δ' ὄσσε ῥόοιο.

ἣ δ' ὑδρηναμένη, καθαρὰ χροῒ εἵμαθ' ἑλοῦσα,
εἰς ὑπερῷ' ἀνέβαινε σὺν ἀμφιπόλοισι γυναιξίν,

Ganz verhaßt; es bleibt ihm noch einer, daß er beherrsche
Dieses hohe Haus und die weiten gesegneten Felder.
Also sprach sie und stillt' ihr den Gram und hemmte die Tränen.

Und sie badete sich und legt' ein reines Gewand an,
Ging hinauf in den Söller, von ihren Mägden begleitet,

Footnotes in running paragraphs

ösen Neigungen zusammen.[d] Methodisch bedeutsam ist aber[e] wieder die G
winnung des Endpunktes <für die Gegenwart>. Dieser[f] muß in einer absolute
und endgültigen Synthese liegen, die eben deshalb nicht aus der natürlichen
ihrem Wesen nach relativistischen> Lebensbewegung [g]stammen oder hervo

a *In A folgt:* wesentlich **b** *A:* Staatsorganismen,

c–*c* *A:* zukünftige und gegenwärtige

d–*d* *A:* Dass er dabei materiell zu einer sehr konservativen, mittelalterlich ständisc
gefärbten und zugleich wieder real-politisch und national gesinnten Staatsauffa
sung kommt, ist eine Sache für sich. Auch dass die Konstruktion der Entwic
lung, die im Grunde immer nur mit einem sehr biologisch getönten Lebensb
griffe arbeitet, kein logisches Fortschrittsprinzip hat, sondern an dessen Stel
sich auf die Vorsehung beruft, ist eine der besonderen Ausführungen des Grun
gedankens. Es gibt hier nicht viel mehr als Spielereien mit völlig unzulängliche
historischen Kenntnissen.

e *A:* erst **f** *A:* Er

g–*g* *A:* mit ihrem unaustilglichen Realismus und Relativismus stammen könne

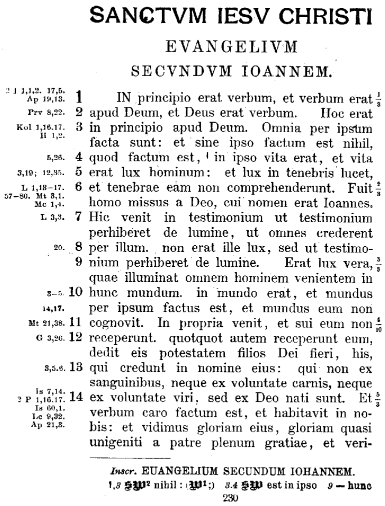
Nested footnotes

<dabei> ist, daß alles das immer nur Einzelentwicklungskreise sind[b] und daß der Fortgang zu einer universalen Verknüpfung all dieser Kreise mit dieser Me

en und Konsequenzen recht interessant, ganz abgesehen von ihrem materiellen Inhalt Hier über das Problem der Geschichtsphilosophie und des Entwicklungsbegriffes Bd I S. V und [c]97. Der[c] alles durchdringende Bewegungsbegriff I 5, 49 f., 30, 179, 251 Universalgeschichte und Vorsehung <I> 79, 147, 95 f. Zusammenfassung von Smith Montesquieu und Burke <I> 86. Mangel eines archimedischen Punktes <für Natur und (offenbarungslose) Geschichte I> 35 f. Die Tendenz des Ganzen [d]III 328: „Den Staat ideenweise (d. h. als Synthese aus Gegensätzen und intuitiv) begreifen heißt ihn für die Gegenwart beseelen, beleben, mit Religion tränken.“[d] [120] <Damit ist auch hier der Zusammenhang der Historie und der gegenwärtigen Kultursynthese scharf behauptet.> Die Ablösung Burkes durch De Bonald, Verm. Schriften[e] I 311 ff. Wichtig und interessant ist[f] der „Briefwechsel mit Gentz <1800–1829“, Stuttgart 1857. – Außerdem hat mir eine lehrreiche Berliner Dissertation von Georg Strauß über „Die Methode A Müllers in der Kritik des 19. und 20. Jahrhunderts“[121] vorgelegen>.

a–***a*** *A:* Romantiker hat dann weiterhin in die Ferne geführt, indische, persische, spanische, französische, englische Geschichte und Geistesentwicklung den Forschern als Gegenstände unterbreitet. Es ist hier nicht möglich, all dem ins einzelne zu folgen und ebenso unmöglich, die mannigfachen Fortwirkungen H. W. Riehl

Tough stuff...



SANCTVM IESV CHRISTI

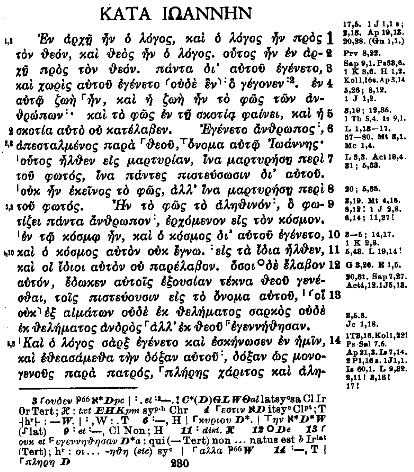
EVANGELIVM

SECVNDVM IOANNEM.

1 J 1,1.2. 17,5. Ap 19,13. 1 IN principio erat verbum, et verbum erat $\frac{1}{3}$
Prv 8,22. 2 apud Deum, et Deus erat verbum. Hoc erat
Kol 1,16.17. H 1,2. 3 in principio apud Deum. Omnia per ipsum
facta sunt: et sine ipso factum est nihil,
5,26. 4 quod factum est, ˡ in ipso vita erat, et vita
3,19; 12,35. 5 erat lux hominum: et lux in tenebris lucet,
L 1,13–17. 57–80. Mt 3,1. Mc 1,4. 6 et tenebrae eam non comprehenderunt. Fuit $\frac{2}{8}$
homo missus a Deo, cui nomen erat Ioannes.
L 3,3. 7 Hic venit in testimonium ut testimonium
perhiberet de lumine, ut omnes crederent
20. 8 per illum. non erat ille lux, sed ut testimo-
9 nium perhiberet de lumine. Erat lux vera, $\frac{3}{8}$
quae illuminat omnem hominem venientem in
3–5. 10 hunc mundum. in mundo erat, et mundus
14,17. per ipsum factus est, et mundus eum non
Mt 21,38. 11 cognovit. In propria venit, et sui eum non $\frac{4}{10}$
G 3,26. 12 receperunt. quotquot autem receperunt eum,
dedit eis potestatem filios Dei fieri, his,
3,5.6. 13 qui credunt in nomine eius: qui non ex
sanguinibus, neque ex voluntate carnis, neque
Is 7,14. 2 P 1,16.17. Is 60,1. Lc 9,32. Ap 21,3. 14 ex voluntate viri, sed ex Deo nati sunt. Et $\frac{5}{3}$
verbum caro factum est, et habitavit in no-
bis: et vidimus gloriam eius, gloriam quasi
unigeniti a patre plenum gratiae, et veri-

Inscr. EUANGELIUM SECUNDUM IOHANNEM.
1,3 **𝔖𝔚**^{2} nihil : (**𝔚**^{1};) *3.4* **𝔖𝔚** est in ipso *9* — **hunc**

230



ΚΑΤΑ ΙΩΑΝΝΗΝ

1,3 Ἐν ἀρχῇ ἦν ὁ λόγος, καὶ ὁ λόγος ἦν πρὸς 1 — 17,5. 1 J 1,1 s; 2,13. Ap 19,13. 20,28. (Gn 1,1.)
τὸν θεόν, καὶ θεὸς ἦν ὁ λόγος. οὗτος ἦν ἐν ἀρ- 2 — Prv 8,22.
χῇ πρὸς τὸν θεόν. πάντα δι' αὐτοῦ ἐγένετο, 3 — Sap 9,1. Ps 33,6. 1 K 8,6. H 1,2.
καὶ χωρὶς αὐτοῦ ἐγένετο ⸀οὐδὲ ἕν⸁ ⁚ ὃ γέγονεν ⁚2. ἐν 4 — Kol 1,16 s. Ap 3,14
αὐτῷ ζωὴ ⸀ἦν, καὶ ἡ ζωὴ ἦν τὸ φῶς τῶν ἀν- — 5,26; 8,12. 1 J 1,2.
θρώπων ⁚· καὶ τὸ φῶς ἐν τῇ σκοτίᾳ φαίνει, καὶ ἡ 5 — 3,19; 12,35. 1 Th 5,4. Is 9,1.
2 σκοτία αὐτὸ οὐ κατέλαβεν. Ἐγένετο ἄνθρωπος ⁚, 6 — L 1,13–17. 57–80. Mt 3,1.
3,3 ἀπεσταλμένος παρὰ ⸀θεοῦ, ⸆ὄνομα αὐτῷ Ἰωάννης· — Mc 1,4.
ˡοὗτος ἦλθεν εἰς μαρτυρίαν, ἵνα μαρτυρήσῃ περὶ 7 — L 3,3. Act 19,4.
τοῦ φωτός, ἵνα πάντες πιστεύσωσιν δι' αὐτοῦ. — 31; 5,33.
ˡοὐκ ἦν ἐκεῖνος τὸ φῶς, ἀλλ' ἵνα μαρτυρήσῃ περὶ 8 — 20; 5,35.
3,3 τοῦ φωτός. Ἦν τὸ φῶς τὸ ἀληθινόν ⁚, ὃ φω- 9 — 3,19. Mt 4,16.
τίζει πάντα ἄνθρωπον ⁚, ἐρχόμενον εἰς τὸν κόσμον. — 8,12! 1 J 2,8. 6,14; 11,27!
ˡἐν τῷ κόσμῳ ἦν, καὶ ὁ κόσμος δι' αὐτοῦ ἐγένετο, 10 — 3–5; 14,17. 1 K 2,8.
4,10 καὶ ὁ κόσμος αὐτὸν οὐκ ἔγνω. ⁚εἰς τὰ ἴδια ἦλθεν, 11 — 5,43. L 19,14!
καὶ οἱ ἴδιοι αὐτὸν οὐ παρέλαβον. ὅσοι °δὲ ἔλαβον 12 — G 3,26. E 1,5.
αὐτόν, ἔδωκεν αὐτοῖς ἐξουσίαν τέκνα θεοῦ γενέ- — 20,31. Sap 7,27. Act 4,12. 1 J 5,13.
σθαι, τοῖς πιστεύουσιν εἰς τὸ ὄνομα αὐτοῦ, ˡ⸀οἳ 13
οὐκ⸁ ἐξ αἱμάτων οὐδὲ ἐκ θελήματος σαρκὸς οὐδὲ — 3,5.6.
ἐκ θελήματος ἀνδρὸς ⸀ἀλλ' ἐκ θεοῦ ⸀ἐγεννήθησαν. — Jc 1,18.
5,3 ˡΚαὶ ὁ λόγος σὰρξ ἐγένετο καὶ ἐσκήνωσεν ἐν ἡμῖν, 14 — 1 T 3,16. Kol 1,22! Ps Sal 7,6.
καὶ ἐθεασάμεθα τὴν δόξαν αὐτοῦ ⁚, δόξαν ὡς μονο- — Ap 21,3. Is 7,14. 2 P 1,16 s. 1 J 1,1.
γενοῦς παρὰ πατρός, ⸀πλήρης χάριτος καὶ ἀλη- — Is 60,1. L 9,32. 2,11! 3,16! 17!

3 ⸀*ουδεν* $\mathfrak{P}^{66}$ ***א*Dpc*** | ⁚ . *et* ⁚2 —. ! ***C*(D)GLWΘal*** latsy^{c}sa Cl Ir Or Tert; ℌ : *txt* ***EHKpm*** sy$^{p.h}$ Chr ***4*** ⸀*εστιν* ***אD*** itsy^{c} Cl^{pt}; T ⊣h^{r}⊢ : —***W***. | ⁚ , W : . T ***6*** ⁚ —, H | ⸀*κυριου* ***D****. | ⸆*ην* ***א*D*W*** (⸉lat) ***9*** ⁚ *et* ⁚ —, Cl Non; H ***11*** ⁚ *dist.* ℌ ***12*** ° ***De*** ***13*** ⸀*ουκ et* ⸀*εγεννηθησαν* ***D*a*** : qui (— Tert) non … natus est *b* Ir^{lat} (Tert); h^{r} : *οι* … *-ηθη* (*sic*) sy^{c} | ⸀*αλλα* $\mathfrak{P}^{66}$***W*** ***14*** ⁚ —, T | ⸀*πληρη* ***D***

230

Contexts

Contexts

- A context is a programmatic entity with its own control flow and local variables.

Contexts

- A context is a programmatic entity with its own control flow and local variables.
- Example: an output context continuously requests material from the main vertical list and insertions. Collections of page matter are then scored (currently this happens using `\brokenpenalty`, `\widowpenalty`, `\clubpenalty`, `\badness` and others).

Contexts

- A context is a programmatic entity with its own control flow and local variables.
- Example: an output context continuously requests material from the main vertical list and insertions. Collections of page matter are then scored (currently this happens using `\brokenpenalty`, `\widowpenalty`, `\clubpenalty`, `\badness` and others).
- The output context thus is coupled with the migration of page material from the vertical list to the current page.

Contexts

- A context is a programmatic entity with its own control flow and local variables.
- Example: an output context continuously requests material from the main vertical list and insertions. Collections of page matter are then scored (currently this happens using `\brokenpenalty`, `\widowpenalty`, `\clubpenalty`, `\badness` and others).
- The output context thus is coupled with the migration of page material from the vertical list to the current page.
- Other contexts may be coupled with other migrations.

Contexts

- A context is a programmatic entity with its own control flow and local variables.
- Example: an output context continuously requests material from the main vertical list and insertions. Collections of page matter are then scored (currently this happens using `\brokenpenalty`, `\widowpenalty`, `\clubpenalty`, `\badness` and others).
- The output context thus is coupled with the migration of page material from the vertical list to the current page.
- Other contexts may be coupled with other migrations.
- For example, a color context would have the current color as a local variable for material migrating to the page and into insertions.

Contexts

- A context is a programmatic entity with its own control flow and local variables.
- Example: an output context continuously requests material from the main vertical list and insertions. Collections of page matter are then scored (currently this happens using `\brokenpenalty`, `\widowpenalty`, `\clubpenalty`, `\badness` and others).
- The output context thus is coupled with the migration of page material from the vertical list to the current page.
- Other contexts may be coupled with other migrations.
- For example, a color context would have the current color as a local variable for material migrating to the page and into insertions.

LuaTEX: Pretty much implementable with coroutines. But: data structure locality? No TEX control flow.

Migrations

- Actions get triggered when objects of a class migrate from one list to another.

Migrations

- Actions get triggered when objects of a class migrate from one list to another.
- Migrations can be penalized.

Migrations

- Actions get triggered when objects of a class migrate from one list to another.
- Migrations can be penalized.
- When different migrations are possible, the combination with the smallest total penalties survives.

Migrations

- Actions get triggered when objects of a class migrate from one list to another.
- Migrations can be penalized.
- When different migrations are possible, the combination with the smallest total penalties survives.
- Line breaking is a special example of penalized breakpoints during the migration of a horizontal into a vertical list.

Migrations

- Actions get triggered when objects of a class migrate from one list to another.
- Migrations can be penalized.
- When different migrations are possible, the combination with the smallest total penalties survives.
- Line breaking is a special example of penalized breakpoints during the migration of a horizontal into a vertical list.

LuaTEX: Provides some hooks/callbacks, but those are not associated with the data itself.

Objects

- are elements of the various horizontal and vertical lists.

Objects

- are elements of the various horizontal and vertical lists.
- can belong to different classes.

Objects

- are elements of the various horizontal and vertical lists.
- can belong to different classes.
- classes can be added as well as extended.

Objects

- are elements of the various horizontal and vertical lists.
- can belong to different classes.
- classes can be added as well as extended.
- objects can have their own contexts for particular migrations.

Objects

- are elements of the various horizontal and vertical lists.
- can belong to different classes.
- classes can be added as well as extended.
- objects can have their own contexts for particular migrations.

LuaTEX: TEX data structures are basically foreign.

Optimization

Global optimization leads to combinatorical explosion of run time.
Countermeasures:

Optimization

Global optimization leads to combinatorical explosion of run time.
Countermeasures:

1. reduction of interdependencies by separated contexts

Optimization

Global optimization leads to combinatorical explosion of run time.
Countermeasures:

1. reduction of interdependencies by separated contexts
2. serialization by tying the optimization to migrations

Optimization

Global optimization leads to combinatorical explosion of run time.
Countermeasures:

1. reduction of interdependencies by separated contexts
2. serialization by tying the optimization to migrations
3. limited backfeed, preferring multiple passes.

Optimization

Global optimization leads to combinatorical explosion of run time.
Countermeasures:

1. reduction of interdependencies by separated contexts
2. serialization by tying the optimization to migrations
3. limited backfeed, preferring multiple passes.
4. make do with less than full optimization.

Disadvantages

Disadvantages

- higher memory impact since decisions need to remain revertible to some degree.

Disadvantages

- higher memory impact since decisions need to remain revertible to some degree.
- higher computational resources because of backtracking

Disadvantages

- higher memory impact since decisions need to remain revertible to some degree.
- higher computational resources because of backtracking
- quite a bit of potential for infinite or almost infinite loops and calculations.

Disadvantages

- higher memory impact since decisions need to remain revertible to some degree.
- higher computational resources because of backtracking
- quite a bit of potential for infinite or almost infinite loops and calculations.
- Programming a full TEX clone on such a platform appears possible, but pointless.

Disadvantages

- higher memory impact since decisions need to remain revertible to some degree.
- higher computational resources because of backtracking
- quite a bit of potential for infinite or almost infinite loops and calculations.
- Programming a full TEX clone on such a platform appears possible, but pointless.
- Decomposition or analysis of several variants can be expensive.

Disadvantages

- higher memory impact since decisions need to remain revertible to some degree.
- higher computational resources because of backtracking
- quite a bit of potential for infinite or almost infinite loops and calculations.
- Programming a full TEX clone on such a platform appears possible, but pointless.
- Decomposition or analysis of several variants can be expensive.

LuaTEX: No relevant hooks or concepts.

Implementation language

Implementation language

- should offer natural expressivity for lists, TeX-typical strings and token lists.

Implementation language

- should offer natural expressivity for lists, TEX-typical strings and token lists.
- should make the required mechanism natively available.

Implementation language

- should offer natural expressivity for lists, TEX-typical strings and token lists.
- should make the required mechanism natively available.
- automatic garbage collection.

Implementation language

- should offer natural expressivity for lists, TeX-typical strings and token lists.
- should make the required mechanism natively available.
- automatic garbage collection.
- need not be a single layer: instead of TeX's Pascal/TeX-macro layering a more tiered concept like C/Scheme/TeX-core/TeX-Macros would be possible.

Implementation language

- should offer natural expressivity for lists, TEX-typical strings and token lists.
- should make the required mechanism natively available.
- automatic garbage collection.
- need not be a single layer: instead of TEX's Pascal/TEX-macro layering a more tiered concept like C/Scheme/TEX-core/TEX-Macros would be possible.
- Problematic: Coroutines. Smalltalk? Ada?

Implementation language

- should offer natural expressivity for lists, TEX-typical strings and token lists.
- should make the required mechanism natively available.
- automatic garbage collection.
- need not be a single layer: instead of TEX's Pascal/TEX-macro layering a more tiered concept like C/Scheme/TEX-core/TEX-Macros would be possible.
- Problematic: Coroutines. Smalltalk? Ada?
- Problematic: I/O (memory for tentative I/O)?

Implementation language

- should offer natural expressivity for lists, TEX-typical strings and token lists.
- should make the required mechanism natively available.
- automatic garbage collection.
- need not be a single layer: instead of TEX's Pascal/TEX-macro layering a more tiered concept like C/Scheme/TEX-core/TEX-Macros would be possible.
- Problematic: Coroutines. Smalltalk? Ada?
- Problematic: I/O (memory for tentative I/O)?
- Combination with low-level languages like C desirable.

Implementation language

- should offer natural expressivity for lists, TEX-typical strings and token lists.
- should make the required mechanism natively available.
- automatic garbage collection.
- need not be a single layer: instead of TEX's Pascal/TEX-macro layering a more tiered concept like C/Scheme/TEX-core/TEX-Macros would be possible.
- Problematic: Coroutines. Smalltalk? Ada?
- Problematic: I/O (memory for tentative I/O)?
- Combination with low-level languages like C desirable.
- Low-level implementation of fast algorithms on custom data structures should be possible

Implementation language

- should offer natural expressivity for lists, TEX-typical strings and token lists.
- should make the required mechanism natively available.
- automatic garbage collection.
- need not be a single layer: instead of TEX's Pascal/TEX-macro layering a more tiered concept like C/Scheme/TEX-core/TEX-Macros would be possible.
- Problematic: Coroutines. Smalltalk? Ada?
- Problematic: I/O (memory for tentative I/O)?
- Combination with low-level languages like C desirable.
- Low-level implementation of fast algorithms on custom data structures should be possible
- Avoidance of unnecessary language features.

Evaluating LuaTEX

- Data structures of TEX are foreign to Lua – Userdata concept helps.

Evaluating LuaTEX

- Data structures of TEX are foreign to Lua – Userdata concept helps.
- TEX's grouping structure does not have a useful equivalent in Lua

Evaluating LuaTEX

- Data structures of TEX are foreign to Lua – Userdata concept helps.
- TEX's grouping structure does not have a useful equivalent in Lua
- catcoded strings have no useful equivalent

Evaluating LuaTEX

- Data structures of TEX are foreign to Lua – Userdata concept helps.
- TEX's grouping structure does not have a useful equivalent in Lua
- catcoded strings have no useful equivalent
- It is unclear how Coroutines and local variables will interplay between TEX and Lua.

Lua Language features

- Lexical scope

Lua Language features

- Lexical scope
- Closures

Lua Language features

- Lexical scope
- Closures
- Coroutines

Lua Language features

- Lexical scope
- Closures
- Coroutines
- Numeric data type is IEEE double, strict superset of both 32bit integers and 14.16 fixpoint numbers

Making use of language features

Control Structures
: Incomplete control structures are not usable. This means that the flow control of the application/format needs to be transferred from TeX to Lua to gain benefits.

Making use of language features

Control Structures
: Incomplete control structures are not usable. This means that the flow control of the application/format needs to be transferred from TEX to Lua to gain benefits.

Lexical scoping
: Making use of scoped variables means that the grouping structure of TEX should not get used. Not feasible with existing formats.

Making use of language features

Control Structures
: Incomplete control structures are not usable. This means that the flow control of the application/format needs to be transferred from TeX to Lua to gain benefits.

Lexical scoping
: Making use of scoped variables means that the grouping structure of TeX should not get used. Not feasible with existing formats.

Data types
: Don't correspond well with TeX's data structures, but then what does? Userdata helps, but grouping?

Making use of language features

Control Structures
: Incomplete control structures are not usable. This means that the flow control of the application/format needs to be transferred from TEX to Lua to gain benefits.

Lexical scoping
: Making use of scoped variables means that the grouping structure of TEX should not get used. Not feasible with existing formats.

Data types
: Don't correspond well with TEX's data structures, but then what does? Userdata helps, but grouping?

Metatables
: Apply for Userdata (one per value). Can be used for operator overloading and other stuff.

Making use of language features

Control Structures
: Incomplete control structures are not usable. This means that the flow control of the application/format needs to be transferred from TEX to Lua to gain benefits.

Lexical scoping
: Making use of scoped variables means that the grouping structure of TEX should not get used. Not feasible with existing formats.

Data types
: Don't correspond well with TEX's data structures, but then what does? Userdata helps, but grouping?

Metatables
: Apply for Userdata (one per value). Can be used for operator overloading and other stuff.

Modules
: possibly nice.

Distributed human workflow

TeX expertise sparse resource

Distributed human workflow

TeX expertise	sparse resource
TeX programming	requires expertise for small tasks

Distributed human workflow

TeX expertise sparse resource

TeX programming requires expertise for small tasks

TeX data structures hell on wheels

Distributed human workflow

TEX expertise	sparse resource
TEX programming	requires expertise for small tasks
TEX data structures	hell on wheels
TEX text processing	catcodes all around

Distributed human workflow

TEX expertise	sparse resource
TEX programming	requires expertise for small tasks
TEX data structures	hell on wheels
TEX text processing	catcodes all around
Consequence	Move complete data and program flow to Lua. Use TEX only for processing fragments.

Refactoring projects

Standalone applications Best chance to change all around, obliterating TEX expert requirements for many tasks.

Refactoring projects

Standalone applications
: Best chance to change all around, obliterating TEX expert requirements for many tasks.

Formats
: Hard to change consistently because data structures and control flow are integrated in TEX and subject to grouping structure.

Refactoring projects

Standalone applications Best chance to change all around, obliterating TeX expert requirements for many tasks.

Formats Hard to change consistently because data structures and control flow are integrated in TeX and subject to grouping structure.

TeX, the Program Rewrite the paragraph optimization framework to be a generally useful mechanism available from Lua?

Refactoring projects

Standalone applications Best chance to change all around, obliterating TEX expert requirements for many tasks.

Formats Hard to change consistently because data structures and control flow are integrated in TEX and subject to grouping structure.

TEX, the Program Rewrite the paragraph optimization framework to be a generally useful mechanism available from Lua?

TEX, the code base Pascal is current “extension language” of TEX. Move part of that to Lua? Data structures?

Interface problems

- Text/Tokenlists

Interface problems

- Text/Tokenlists
- Tokenlists/Lua Code

Interface problems

- Text/Tokenlists
- Tokenlists/Lua Code
- Strings/TEX Code

Interface problems

- Text/Tokenlists
- Tokenlists/Lua Code
- Strings/TEX Code
- Boxes/Node list manipulation

Programming features shortlist

Feature	Lua	TeX
Scope	lexical with closures	sickly dynamic
Code execution	byte code	token list processing
Loops	nestable scopes	unnested, mouth/stomach chimera
List access	$O(lg(lg(n)))$	$O(n)$ if you are really good
Indexing	reasonable Hash, exception for numeric	bad hash particularly for numeric data
Coroutines	elegant	Coroutines?
Language design	minimalist	Language design?

And now for something completely different(?)

Ampulex compressa is a wasp that has evolved to tackle roaches, insert a stinger into their brains and disable their escape reflexes. This lets the wasp use the roach's antennae to steer the roach to its lair, where it can lay its egg in it.
The roach is no longer able to move on its own. When the egg layed on the underside of the roach hatches, the larva enters the host roach and feeds on its organs for eight days. It makes a cocoon and pupates. After 4 weeks it emerges from the roach as a full-grown wasp.
(use a search engine on "zombie cockroach").