the setl project - master catalog

part 1 - major documents.
part 2 - SETL newsletters.
part 3 - LITTLE newsletters.
part 4 - the SETL algorithms library.
part 5 - the SETL test packages.

subscribers, please note:

up to five back issues of SETL or LITTLE newsletters are available free on request in single copies. those wishing to be added to the regular mailing list maintained for these newsletters should write to:

SETL Project
Courant Institute of Mathematical Sciences
New York University
251 Mercer Street
New York, New York 10012

enclosing a check for ten dollars for a one year subscription which should be made out to New York University.

Certain of our longer documents, listed on the next page, are available direct from our publications co-ordinator. Write to:

Lenora Greene, SETL Publications Co-ordinator
Courant Institute of Mathematical Sciences
251 Mercer Street
New York, New York 10012

   J. Schwartz

   part 1: generalities.
   part 2: the SETL language, and examples of its use.

   (revised) June 1975  xii + 675 pp

   part 3: extended facilities of the SETL language.
   (to appear)
2. A SETL primer. H. Mullish
M. Goldstein
A step-by-step tutorial with over 100 illustrative
programmettes.
June 1973 v + 201 pp

3. The SETL run-time library.
This is the run-time support system for SETL. It is
written in LITTLE and is well documented internally.
It supports all of the main set-theoretic primitives
of SETL and is of prime interest to those wishing to
develop or modify the SETL system. The run-time
library is available in machine readable form.

4. Courant computer sciences reports.

1. ASL: a proposed variant of SETL.
H. Warren
(out of print) May 1973 xi+326 pp

2. A metalanguage for expressing grammatical restrictions in
nodal spans parsing of natural language.
J. Hoobs
January 1974 266 pp

3. Type determination for very high level languages.
A. Tenenbaum
(out of print) October 1974 v+171 pp

W. Gewirtz
October 1974 60 pp

5. Operating system specification using very high level dictions.
R. Markstein
(out of print) June 1973 152 pp

6. Directions in artificial intelligence: natural language
processing.
R. Grishman (ed.)
(out of print) August 1975 109 pp
|    | (out of print) | R. Grishman |
|    | C. Weiman |
| 10. | A hierarchical technique for mechanical theorem proving and its application to programming language design. | November 1976 | 172 pp |
|    | (out of print) | N. Rubin |
|    | J. Hoops |
|    | S. Rosenschein |
| 12. | Correct program technology / extensibility of verifiers. | September 1977 | 146 pp |
|    | M. Davis |
|    | J. Schwartz |
|    | E. Deak (appendix) |
|    | C. Semeniuk |
|    | J. Fabri |
|    | S. Liu |
|    | R. Paige |
| 16. | On the Complexity of the Satisfiability Problem. | October 1979 | 85 pp |
|    | A. Goldberg |
| 17. | A Design for Optimizations of the Bitvectorsing Class. | September 1979 | 117 pp |
|    | J. Schwartz |
|    | M. Sharir |
| 18. | Automatic Discovery of Heuristics for Nondeterministic Programs from Sample Execution Traces. | September 1979 | 172 pp |
|    | S. Stolfo |
|    | B. Loerinc |
20. The Language REFAI - The Theory of Complication and Metasystem Analysis. V. Turchin
February 1980 245 pp
5. The LITTLE system manual.

The system manual for the LITTLE language addresses the following needs: a system programmer's reference manual giving details of compiler structure, installation and maintenance; an algorithmic description of the compiler structure; a guide to extending the compiler to support new language features; a case study of an actual implementation and of the problems encountered in constructing machine independent software; a pragmatic reference document for a compiler-writing course.


Subscribers who want complete, unbound back issues of newsletters should write to SETL Project at the address given on page 1, enclosing a check made out to New York University to cover cost as follows:

- SETL newsletters - part 2 of this catalog.
  price $50.00
- LITTLE newsletters - part 3 of this catalog.
  price $20.00

7. Various machine readable information.

Those interested in acquiring any of the following documents in machine readable form should address an inquiry concerning price to SETL Project at the address given on page 1.

- SETL newsletter 42 - detailed specifications of certain SETL operations described in part 6 of this catalog.

Following is an inclusive, up-to-date list of the SETL newsletters. Many of our earliest newsletters are not relevant to our current goals and have therefore been marked: obsolete near the right hand margin of the pages on which they appear. Obsolete newsletters are available by special request only - they will not be provided in standard orders for back issues.

1. BALM-SETL -- a simple implementation of SETL.
   November 1970
   M. Harrison
   obsolete

2. No longer available.
Part 1 - major documents.

   J. Shields
   November 1970 6 pp   obsolete

4. An APL version of Peter Markstein's McKeeman table routine.
   P. Markstein
   November 1970 2 pp   obsolete

5. Miscellaneous algorithms written in SETL.
   J. Schwartz
   November 1970 7 pp   obsolete

6. A revised SETL version of the McKeeman parse.
   P. Markstein
   November 1970 3 pp   obsolete

   J. Shields
   November 1970 8 pp   obsolete

8. Additional miscellaneous SETL algorithms.
   J. Schwartz
   November 1970 4 pp   obsolete

9. Implementation and Language design.
   M. Harrison
   December 1970 5 pp   obsolete

10. A sorting algorithm.
    K. Maly
    December 1970 3 pp   obsolete

11. Modifications and extensions for SETL, part 3.
    J. Shields
    December 1970 4 pp   obsolete

12. Recapitulation of the basic parts of the SETL language.
    J. Schwartz
    January 1971 14 pp   obsolete

13. Additional miscellaneous algorithms.
    J. Schwartz
    January 1971 7 pp   obsolete

    J. Schwartz
    January 1971 3 pp

15. A proposed SETL implementation plan through the end of the
    bootstrap phase.
    J. Schwartz
    February 1971 2 pp   obsolete

16. SETL 64-character set -- 49-character set / 025 keypunch -- cdc
1600 84-character set / O23 keypunch. K. Maly
February 1971 2 pp obsolete

17. No longer available.

18. Preliminary specification of BALMSETL conventions.
February 1971 2 pp obsolete

19. Lexical description of SETL.
February 1971 5 pp obsolete

20. BALMSETL users guide (in brief).
March 1971 4 pp obsolete

21. An outside review: comments on the SETL draft.
April 1971 8 pp anon. (publisher)

22. Some small and large language extensions for consideration.
April 1971 4 pp J. Schwartz

April 1971 3 pp obsolete J. Schwartz

24. Description of a register allocation algorithm.
April 1971 8 pp K. Kennedy

25. A print routine.
April 1971 3 pp B. Loerinc

26. The currently specified form of SETL from a more fundamental
May 1971 8 pp point of view. J. Schwartz

27. Code for the postparse setup procedure (postparse metalanguage
May 1971 15 pp obsolete analysis).

28. An algorithm for common subexpression elimination and code
May 1971 10 pp motion. K. Kennedy

29. Some issues connected with subroutine linkage.
May 1971 3 pp J. Schwartz

30. Sinister calls.
Part 1 - major documents.

31. An additional preliminary remark on the importance of "object types" for SETL, with some reflections on the motion of "data structure language".
   J. Schwartz

32. Hyper-SETL procedural languages.
   J. Schwartz

33. What is programming.
   J. Schwartz

34. Syntax revisions in preparation for implementation.
   J. Schwartz

35. A new form for the IF statement.
   J. Shields

36. No longer available.

   P. Owens

38. An algorithm for live-dead analysis including node-splitting for irreducible program graphs.
   K. Kennedy

39. More detailed suggestions concerning "data strategy" elaborations for SETL.
   J. Schwartz

40. No longer available.

41. Additional planning detail for the current and next phase of SETL implementation.
   J. Schwartz

42. Revised conventions concerning tuples.
   J. Schwartz

43. A parsing scheme for FORTRAN.
   S. Gruber

44. Comprehensive SETL specifications.
   K. Maity
Part 1 - major documents.

45. Semi-local SETL optimization.
   July 1971 11 pp
   D. Shields

46. Generalized nodal scan parse routine - preliminary draft.
   July 1971 12 pp
   J. Schwartz

47. An outline for a parsing scheme for SETL.
   July 1971 3 pp
   K. Maly

48. Toward a documentation of the string project's program for parsing English sentences.
   August 1971 19 pp
   J. Dubs

49. Specification of the SETL run time library (revision 2).
   April 1973 139 pp
   H. Warren

50. A three-phase parsing scheme for SETL.
    September 1971 3 pp
    K. Maly

51. No longer available.

52. Comments on SETL.
    September 1971 8 pp
    J. Earley

53. SETL to LITTLE translator: a first look.
    September 1971 26 pp
    H. Warren

    September 1971 5 pp
    S. Gruber

55. SETL suggestions and questions.
    September 1971 5 pp
    S. Finkelstein

56. Additional comments on some basic SETL operations.
    September 1971 4 pp
    J. Earley

56a. More comments on SETL.
     October 1971 5 pp
     J. Earley

56b. More SETL comments.
     J. Earley
<table>
<thead>
<tr>
<th>Document Title</th>
<th>Date</th>
<th>Pages</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>57. Minimizing copying in SETL: preliminary observations.</td>
<td>October 1971</td>
<td>10</td>
<td>H. Warren</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>obsolete</td>
</tr>
<tr>
<td>58. Phase one of the SETL compiler.</td>
<td>October 1971</td>
<td>21</td>
<td>K. Maly</td>
</tr>
<tr>
<td>60. SETL compiled code: calls to SETL procedures.</td>
<td>November 1971</td>
<td>48</td>
<td>H. Warren</td>
</tr>
<tr>
<td>61. Syntactic structure of SETL.</td>
<td>November 1971</td>
<td>14</td>
<td>K. Maly</td>
</tr>
<tr>
<td>63. The SETL print routine.</td>
<td>January 1972</td>
<td>5</td>
<td>G. Fisher</td>
</tr>
<tr>
<td>64. SETL compiler with elaborated data structures.</td>
<td>January 1972</td>
<td>33</td>
<td>K. Maly</td>
</tr>
<tr>
<td>65. Some notational suggestions.</td>
<td>February 1972</td>
<td>2</td>
<td>R. Bonic</td>
</tr>
<tr>
<td>66. BALMSETL user's manual version 1.0 (revised 9/72).</td>
<td>February 1972</td>
<td>71</td>
<td>E. Milgrom</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>obsolete</td>
</tr>
<tr>
<td>67. Data structures of the SETL compiler from the LITTLE version.</td>
<td>February 1972</td>
<td>16</td>
<td>K. Maly</td>
</tr>
<tr>
<td>68. Some thoughts on efficient programming in SETL.</td>
<td>October 1972</td>
<td>5</td>
<td>S. Brown</td>
</tr>
<tr>
<td>69. The SETL project - master catalog (revised 2/73).</td>
<td>February 1973</td>
<td>34</td>
<td>R. Abes</td>
</tr>
<tr>
<td>70. The SETL user's manual (revised 1/75).</td>
<td></td>
<td></td>
<td>J. Schwartz</td>
</tr>
<tr>
<td>Document Number</td>
<td>Title</td>
<td>Authors</td>
<td>Date</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------------------------------------------------------------</td>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>71.</td>
<td>Deducting the logical structure of objects occurring in SETL programs.</td>
<td>J. Schwartz</td>
<td>April 1972</td>
</tr>
<tr>
<td>72.</td>
<td>An introductory explanation of SETL: a status review and profile of the SETL user group.</td>
<td>J. Shields</td>
<td>April 1972</td>
</tr>
<tr>
<td>73.</td>
<td>User's guide to the SETL run-time library. (revision 3 - May 1973)</td>
<td>K. Maly</td>
<td>April 1972</td>
</tr>
<tr>
<td>74.</td>
<td>Project plan for first stage of implementation. (partial translation from the russian)</td>
<td>L. Gorodnaya, D. Lewin</td>
<td>May 1972</td>
</tr>
<tr>
<td>75.</td>
<td>Some thoughts on the use of BALM to implement SETL. (this is also BALM bulletin no. 13)</td>
<td>E. Milgrom</td>
<td>June 1972</td>
</tr>
<tr>
<td>76.</td>
<td>Semantic definition matters.</td>
<td>J. Schwartz, G. Jennings</td>
<td>May 1973</td>
</tr>
<tr>
<td>77.</td>
<td>Transferring SETLB to other machines.</td>
<td>J. Schwartz</td>
<td>September 1972</td>
</tr>
<tr>
<td>78.</td>
<td>Executing BALM and SETLB at NYU Courant.</td>
<td>R. Paige</td>
<td>September 1972</td>
</tr>
<tr>
<td>79.</td>
<td>No longer available.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80.</td>
<td>Algorithms in the SETLB test package.</td>
<td>K. Curtis</td>
<td>September 1972</td>
</tr>
<tr>
<td>81.</td>
<td>Memory size of SETLB runs.</td>
<td>J. Schwartz, S. Brown</td>
<td>September 1972</td>
</tr>
<tr>
<td>82.</td>
<td>Timing comparison between SETLB and FORTRAN</td>
<td>E. Desautels</td>
<td>October 1972</td>
</tr>
<tr>
<td>84.</td>
<td>Plan for a library of algorithms.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
25. Estimate of minimum running size for the next SETL system (revision 1).  
November 1972 18 pp  J. Schwartz

26. Proposal for a temporary, but easily implemented, software paging system.  
November 1972 7 pp  J. Schwartz

27. Workplan for the next phase of SETL implementation.  
November 1972 7 pp  J. Schwartz

November 1972 2 pp  J. Schwartz

29. User information for lexical scan setup package.  
November 1972 2 pp  E. Suth

December 1972 18 pp  J. Schwartz

31. A grammarless parse and a related method of retrieval by similarity.  
December 1972 24 pp  J. Schwartz

32. Some experiments with SETL programs.  
January 1973 9 pp  K. Curtis

33. A note on optimization and programming style in SETL.  
January 1973 2 pp  K. Curtis

34. An algorithm to represent a collection of sets as intervals (on a line).  
January 1973 9 pp  G. Jennings

35. Generalized nodal span parse routine, corrected version.  
January 1973 10 pp  Y. Feinroth

36. Pointers and 'very high level languages'.  
January 1973 3 pp  N. Minsky

37. SETL extensions for operating system description.  
February 1973 24 pp  P. Markstein
<table>
<thead>
<tr>
<th>No.</th>
<th>Title</th>
<th>Author(s)</th>
<th>Date</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>98.</td>
<td>Reflections on P. Markstein's newsletter on SETL extensions for operating system description.</td>
<td>J. Schwartz</td>
<td>January 1973</td>
<td>9 pp</td>
</tr>
<tr>
<td>99.</td>
<td>Paging, the quick and dirty way. (this is also Balm Bulletin No. 21)</td>
<td>S. Brown</td>
<td>January 1973</td>
<td>4 pp</td>
</tr>
<tr>
<td>100.</td>
<td>Making SRTL debugging runs.</td>
<td>H. Warren</td>
<td>February 1973</td>
<td>11 pp</td>
</tr>
<tr>
<td>101.</td>
<td>How to program if you must (the SRTL style).</td>
<td>R. Bonic</td>
<td>March 1973</td>
<td>15 pp</td>
</tr>
<tr>
<td>103.</td>
<td>Preliminary plan for Balm-to-LITTLE translator.</td>
<td>J. Schwartz</td>
<td>April 1973</td>
<td>8 pp</td>
</tr>
<tr>
<td>104.</td>
<td>An algorithm to represent a collection of sets as a direct product of intervals on the line.</td>
<td>G. Jennings</td>
<td>March 1973</td>
<td>9 pp</td>
</tr>
<tr>
<td>105.</td>
<td>A SRTL program for a basic block optimizer and an extended basic block optimizer.</td>
<td>S. Harateck</td>
<td>April 1973</td>
<td>11 pp</td>
</tr>
<tr>
<td>106.</td>
<td>User variation of the semantics of function and subroutine invocation.</td>
<td>G. Jennings</td>
<td>May 1973</td>
<td>3 pp</td>
</tr>
<tr>
<td>107.</td>
<td>Linear function test replacement.</td>
<td>K. Kennedy</td>
<td>May 1973</td>
<td>3 pp</td>
</tr>
<tr>
<td>108.</td>
<td>APL - SRTL, an extension of SRTL achieved from user varied semantics.</td>
<td>G. Jennings</td>
<td>May 1973</td>
<td>34 pp</td>
</tr>
<tr>
<td>109.</td>
<td>Faster execution for the LITTLE based balm system.</td>
<td>S. Brown</td>
<td>July 1973</td>
<td>4 pp</td>
</tr>
<tr>
<td>111.</td>
<td>Global dead computation elimination.</td>
<td>K. Kennedy</td>
<td>August 1973</td>
<td>8 pp</td>
</tr>
<tr>
<td>112.</td>
<td>An algorithm to compute compacted use-definition chains.</td>
<td>K. Kennedy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Document Number</td>
<td>Title</td>
<td>Date</td>
<td>Pages</td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------------------------------------------------------------</td>
<td>------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>113</td>
<td>LITTLE code generation from the BALM compiler.</td>
<td>August 1973</td>
<td>9 pp</td>
<td></td>
</tr>
<tr>
<td>114</td>
<td>A SETLB to publication SETL translator.</td>
<td>August 1973</td>
<td>7 pp</td>
<td></td>
</tr>
<tr>
<td>115</td>
<td>A SETL representation of the Maryland GRAAL graph-manipulation language.</td>
<td>August 1973</td>
<td>32 pp</td>
<td></td>
</tr>
<tr>
<td>116</td>
<td>Catalog of SETL(c) newsletters as of July 30, 1973.</td>
<td>August 1973</td>
<td>5 pp</td>
<td></td>
</tr>
<tr>
<td>117</td>
<td>A static debugging system for LITTLE.</td>
<td>October 1973</td>
<td>12 pp</td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>Revised and extended algorithms for deducing the types of objects occurring in SETL programs.</td>
<td>October 1973</td>
<td>21 pp</td>
<td></td>
</tr>
<tr>
<td>119</td>
<td>A suggested generalization and revision of the SETL compound operator form.</td>
<td>October 1973</td>
<td>2 pp</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>A general-recursive extension of functional application and its uses.</td>
<td>December 1973</td>
<td>3 pp</td>
<td></td>
</tr>
<tr>
<td>121</td>
<td>An algorithm to determine the identity of SETL run-time objects.</td>
<td>January 1974</td>
<td>15 pp</td>
<td></td>
</tr>
<tr>
<td>122</td>
<td>More local and semi-local SETL optimisations.</td>
<td>January 1974</td>
<td>6 pp</td>
<td></td>
</tr>
<tr>
<td>122A</td>
<td>A few peephole optimisations applicable to iterators.</td>
<td>July 1974</td>
<td>2 pp</td>
<td></td>
</tr>
<tr>
<td>122B</td>
<td>Still more miscellaneous optimisations.</td>
<td>July 1974</td>
<td>6 pp</td>
<td></td>
</tr>
<tr>
<td>123</td>
<td>Variable subsumption with constant folding.</td>
<td>February 1974</td>
<td>13 pp</td>
<td></td>
</tr>
<tr>
<td>124</td>
<td>The VERS2 language of J. Earley considered in relation to SETL.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Part 1 - major documents.

January 1974 5 pp  E. Schonberg

125. Schaefer's node splitting algorithm.

February 1974 14 pp  K. Kennedy

126. A SETLB specification of EDIT.

February 1974 112 pp  M. Brenner

127. Edge-listing data-flow algorithms.

March 1974 9 pp  K. Kennedy

128. A LITTLE written translator from SETL to LITTLE.

April 1974 9 pp  E. Schonberg

129. More on SETL in a data-base environment.

May 1974 43 pp  J. Schwartz

130. Deducing relationships of inclusion and membership in SETL
    programs.

May 1974 36 pp  J. Schwartz

130a. Estimates from below the domain of a mapping.

August 1974 7 pp  J. Schwartz

130b. The use of equalities in the deduction of
    inclusion/membership relations.

May 1975 12 pp  J. Schwartz

131. More on copy optimisation of SETL programs.

June 1974 51 pp  J. Schwartz

132. Some optimisations using type information.

June 1974 3 pp  J. Schwartz

133. A higher level control diction.

June 1974 5 pp  J. Schwartz

133A. Additional pursue block examples.

June 1974 5 pp  J. Schwartz

133b. General comments on high level dictions, and specific sugges-
    tions concerning 'converge! iterators and some related dictions.

January 1975 18 pp  J. Schwartz
134. Inter-procedural optimisation.  
    July 1974  24 pp

135. Introductory lecture at the June 28 "informal optimisation symposium".  
    July 1974  14 pp

135a. Structureless programming -or- the notion of "rubble" and the reduction of programs to rubble.  
    July 1974  25 pp

135b. Additional thoughts on "language level" and optimisation.  
    January 1975  2 pp

135c. On the genesis of complex programs.  
    August 1975  4 pp

135d. A framework for certain kinds of high-level optimisation.  
    July 1974  4 pp

135e. Additional thoughts concerning automatic data structure choice.  
    July 1974  12 pp

136. On Jay Earley's "method of iterator inversion".  
    in progress  R. Paige

136a. Optimisation by set suppression.  
    August 1974  3 pp

136b. Updating the lower bound of a set of integers in set-theoretic strength reduction.  
    January 1976  4 pp

140. Use-use chaining as a technique in typefinding.  
    September 1974  5 pp

141. Reflections on some very high level dictions having an english / "automatic programming" flavor.  
    January 1975  18 pp

142. What programmers should know.  
    January 1975  5 pp
<table>
<thead>
<tr>
<th>#</th>
<th>Title</th>
<th>Author(s)</th>
<th>Date</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>143</td>
<td>&quot;Arguments from use&quot; in the proof of relationships of inclusion and membership.</td>
<td>J. Schwartz</td>
<td>February 1975</td>
<td>2 pp</td>
</tr>
<tr>
<td>144</td>
<td>Interprocedural live-dead analysis.</td>
<td>J. Schwartz</td>
<td>February 1975</td>
<td>3 pp</td>
</tr>
<tr>
<td>145</td>
<td>GYVE-oriented inter-process coordination and control structures for an extended SETL (SETLG).</td>
<td>J. Schwartz</td>
<td>March 1975</td>
<td>17 pp</td>
</tr>
<tr>
<td>146</td>
<td>Adaption of GYVE/SETLG to distributed networks of computers.</td>
<td>J. Schwartz</td>
<td>March 1975</td>
<td>5 pp</td>
</tr>
<tr>
<td>147</td>
<td>A syntactic construct useful for checking parameters.</td>
<td>J. Schwartz</td>
<td>March 1975</td>
<td>4 pp</td>
</tr>
<tr>
<td>148</td>
<td>Technical and human factors improvements for the fully compiled SETL system.</td>
<td>E. Schonberg</td>
<td>April 1975</td>
<td>7 pp</td>
</tr>
<tr>
<td></td>
<td></td>
<td>a. Stein</td>
<td></td>
<td></td>
</tr>
<tr>
<td>149</td>
<td>Conventions allowing other languages to be used within GYVE; files; memory hierarchy questions; some suggestions for GYVE extensions.</td>
<td>J. Schwartz</td>
<td>April 1975</td>
<td>28 pp</td>
</tr>
<tr>
<td>150</td>
<td>What constitutes process in programming.</td>
<td>J. Schwartz</td>
<td>May 1975</td>
<td>5 pp</td>
</tr>
<tr>
<td>151</td>
<td>Additional considerations concerning semi-automatic data structure choice.</td>
<td>J. Schwartz</td>
<td>July 1975</td>
<td>37 pp</td>
</tr>
<tr>
<td>152</td>
<td>An alternative design for a &quot;MIDL&quot;-level language.</td>
<td>J. Schwartz</td>
<td>July 1975</td>
<td>4 pp</td>
</tr>
<tr>
<td>153</td>
<td>The significance of &quot;backtracking&quot;, and its cost.</td>
<td>J. Schwartz</td>
<td>July 1975</td>
<td>20 pp</td>
</tr>
<tr>
<td>154</td>
<td>Timing considerations for the SETL translator system.</td>
<td>E. Schonberg</td>
<td>July 1975</td>
<td>6 pp</td>
</tr>
<tr>
<td>155</td>
<td>Intermediate result recording and other techniques for optimizing recursions and backtrack programs.</td>
<td>J. Schwartz</td>
<td>October 1975</td>
<td>26 pp</td>
</tr>
</tbody>
</table>
156. The next phase of our work. J. Schwartz
   August 1975 4 pp

157. "Whenever" dictions. J. Schwartz
   August 1975 24 pp

158. Implementation of reference counts in the SETL system. E. Schonberg
   October 1975 4 pp

159. On the "base form" of algorithms. J. Schwartz
   November 1975 15 pp

160. An algebra of program events potentially useful in a debugging language. J. Schwartz
   November 1975 5 pp

161. Specifications for a new optimizer-oriented SETL front end. A. Brand
   December 1975 13 pp

162. Improved target code forms available in the presence of global information concerning a SETL program. January 1976 20 pp

163. Recognizing comparability graphs in SETL. M. Golumbic
   March 1976 9 pp

164. Copy optimization in SETL. J. Schwartz
   April 1976 3 pp

164a. "Copy on assignment" optimization in SETL. R. Dewar
   April 1976 9 pp

165. A simple criterion for avoiding basing errors. J. Schwartz
   April 1976 1 pp

166. An easy scheme for incorporating backtracking into the new SETL implementation. J. Schwartz R. Dewar
   April 1976 13 pp

167. A variant SETL implementation incorporating "whenever" dictions. J. Schwartz
   April 1976 7 pp
168. The trouble with triples.  
April 1976 7 pp  R. Dewar

169. Some changes to the SETL language in preparation for the optimizer implementation.  
April 1975 14 pp  R. Dewar  A. Grand  E. Schonberg

170. Provisional plan for the SETL optimizer interface.  
April 1975 3 pp  R. Dewar  A. Grand  E. Schonberg  L. Vanek

171. More on basings.  
April 1976 29 pp  J. Schwartz

171a. Still more on basings.  
December 1976 10 pp  J. Schwartz

171b. Remark on the implementation of the basing scheme.  
December 1976 3 pp  J. Schwartz

171c. Implementation of base assignments.  
April 1977 9 pp  A. Grand

172. A case statement for SETL.  
May 1976 3 pp  R. Dewar  A. Grand  E. Schonberg  L. Vanek

173. Simplifying and extending the SETL type calculus.  
April 1976 14 pp  L. Vanek

174. Relaxing of basing restrictions.  
July 1976 3 pp  R. Dewar

175. More on copy optimization.  
July 1976 4 pp  S. Liu  E. Schonberg

176. A coarser, but simpler and considerably more efficient copy optimization technique.  
J. Schwartz
<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
<th>Date</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measurement utilities for the optimized SETL system.</td>
<td>J. Schwartz</td>
<td>August 1976</td>
<td>5 pp</td>
</tr>
<tr>
<td>Motion of range checks out of loops; optimization of integer arithmetic.</td>
<td>J. Schwartz</td>
<td>August 1976</td>
<td>2 pp</td>
</tr>
<tr>
<td>Dynamic multiple member basings.</td>
<td>S. Liu</td>
<td>October 1976</td>
<td>6 pp</td>
</tr>
<tr>
<td>Uncovering profitable basing relations.</td>
<td>S. Liu, E. Schonberg</td>
<td>February 1977</td>
<td>10 pp</td>
</tr>
<tr>
<td>A reformulation of value-flow analysis.</td>
<td>W. Tsui</td>
<td>March 1977</td>
<td>33 pp</td>
</tr>
<tr>
<td>Linkage conventions for the SETL optimizer.</td>
<td>L. Vanek</td>
<td>November 1976</td>
<td>8 pp</td>
</tr>
<tr>
<td>Extending the notion of basing.</td>
<td>R. Dewar</td>
<td>March 1977</td>
<td>2 pp</td>
</tr>
<tr>
<td>Using output form the SETL copy optimizer.</td>
<td>L. Vanek</td>
<td>March 1977</td>
<td>6 pp</td>
</tr>
<tr>
<td>Syntax and semantics of a restricted backtrack implementation.</td>
<td>R. Dewar, J. Schwartz</td>
<td>April 1977</td>
<td>9 pp</td>
</tr>
<tr>
<td>On inter-procedural flow analysis.</td>
<td>M. Sharir</td>
<td>April 1977</td>
<td>14 pp</td>
</tr>
<tr>
<td>More on inter-procedural data flow analysis.</td>
<td>M. Sharir</td>
<td>May 1977</td>
<td>8 pp</td>
</tr>
</tbody>
</table>
188. A limited form of common subexpression elimination for SETL programs.  
   May 1977 15 pp  
   L. Vanek

190. The implementation of backtracking.  
   May 1977 14 pp  
   A. Grand

191. More language changes.  
   May 1977 4 pp  
   A. Grand

192. 6600, 370, and PUMA microcode nubbins.  
   May 1977 13 pp  
   A. Grand  
   J. Schwartz  
   R. Kenner

193. On a static scheme to find procedure variables.  
   May 1977 5 pp  
   M. Sharir

194. Nondeterminism, backtracking, and pattern matching in SETL.  
   June 1977 15 pp  
   S. Rapps

195. An algorithm for copy optimization, based on NL 176.  
   July 1977 13 pp  
   M. Sharir

196. Current state of the SETL implementation.  
   August 1977 2 pp  
   A. Grand

197. Some comments on extending code motion and expression availability algorithms for the SETL optimizer.  
   September 1977 6 pp  
   M. Sharir

198. String primitives.  
   January 1978 5 pp  
   R. Dewar

199. Non-propagation of errors - a modified type-finding algorithm.  
   January 1978 5 pp  
   M. Sharir

200. Possible additional reps for the SETL system.  
   February 1978 7 pp  
   J. Schwartz
201. On compaction on re-paths.  
   M. Sharir  
   February 1978  3 pp

   H. Lewis  
   July 1978  3 pp

203. A simplified approach to automatic data structure choice.  
   M. Sharir  
   January 1978  10 pp

204. Tarjan's fast interval finding algorithm.  
   M. Sharir  
   March 1978  12 pp  
   J. Schwartz

205. On disjointness detection in automatic data structure choice.  
   M. Sharir  
   July 1978  8 pp

206. On name splitting in SETL optimization.  
   A. Grand  
   M. Sharir  
   February 1978  16 pp

207. A second simplified approach to automatic data structure choice.  
   M. Sharir  
   March 1978  21 pp

208. A few cautionary notes on the convergence of iterative data-flow analysis algorithms.  
   M. Sharir  
   April 1978  9 pp

209. Automatic data structure choice.  
   E. Schonberg  
   March 1978  21 pp

210. Remarks on debugging.  
   J. Schwartz  
   February 1979  8 pp

211. The SETL Character Set - The Final Decisions.  
   R. Dewar  
   A. Grand  
   March 1979  4 pp

212. A Note on Program Genesis.  
   J. Schwartz  
   September 1979  4 pp

1. I/O conventions and proposal; quoted strings; octal constants; user information for improved macroprocessor.
<table>
<thead>
<tr>
<th>No.</th>
<th>Title</th>
<th>Author</th>
<th>Date</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.</td>
<td>No longer available.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>Possible future extensions to LITTLE.</td>
<td>J. Schwartz</td>
<td>November 1971</td>
<td>16</td>
</tr>
<tr>
<td>5.</td>
<td>User information concerning the LITTLE-to-FORTRAN translator.</td>
<td>J. Schwartz</td>
<td>November 1971</td>
<td>3</td>
</tr>
<tr>
<td>6.</td>
<td>No longer available.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>LITTLE for minicomputers.</td>
<td>T. Stuart</td>
<td>March 1972</td>
<td>25</td>
</tr>
<tr>
<td>8.</td>
<td>No longer available.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td>Some suggestions for simplifying the preparation of SETL and LITTLE text: keyboard and lexical macros.</td>
<td>J. Shields</td>
<td>March 1972</td>
<td>6</td>
</tr>
<tr>
<td>10.</td>
<td>Interspersing macros.</td>
<td>J. Schwartz</td>
<td>April 1972</td>
<td>9</td>
</tr>
<tr>
<td>11.</td>
<td>Input / output statements for LITTLE.</td>
<td>R. Abes H. Warren</td>
<td>April 1972</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15.</td>
<td>No longer available.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16.</td>
<td>Some timing statistics for LITTLE.</td>
<td>D. Shields</td>
<td>October 1972</td>
<td>10</td>
</tr>
<tr>
<td>17.</td>
<td>Test packages for the LITTLE compiler.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
18. A new array optimization for basic blocks.  
   J. Schwartz  
   November 1972  4 pp

19. No longer available.

20. Remarks on the structure of the SETL run time library.  
   D. Shields  
   November 1972  5 pp

21. Some proposals for improving the accessibility of the LITTLE compiler.  
   J. Shields  
   December 1972  9 pp

22. Examples of LITTLE-generated code.  
   D. Shields  
   December 1972  5 pp

23. Namesets: a new way to handle global variables in LITTLE.  
   D. Shields  
   January 1973  6 pp

24. Proposals for the next stage of LITTLE development.  
   D. Shields  
   March 1973  6 pp

25. Proposed extensions to LITTLE.  
   D. Shields  
   June 1973  21 pp

   L. Chernobrod  
   August 1973  5 pp

27. Multiple word items in LITTLE.  
   R. Abes  
   October 1973  13 pp

28. An intermediate language for the LITTLE compiler.  
   S. Brown  
   October 1973  6 pp

29. A medium-level semantic environment based on LITTLE.  
   J. Schwartz  
   September 1973  20 pp

30. Interrupt handling facilities in LITTLE.  
   J. Schwartz  
   December 1973  17 pp

31. Representation of BALM in LITTLE.
32. Interrupt handling in LITTLE: possible revisions.
   P. Shaw

February 1974
20 pp

32a. Realisation of the LITTLE interrupt system described in newsletters
      nos. 30 and 32.
   C. McDonald

January 1975
79 pp

33. Guide to the LITTLE language.
   J. Shields

March 1974
90 pp

34. Input/output statements for LITTLE.
   (this will replace LITTLE newsletter 11.) R. Abes
   to appear shortly.

   I. Chakravarty
   H. Jacobs
   M. Marks
   E. Mcgovern

January 1975
39 pp

35b. LITTLE code generator for the IBM 1130.
   R. Aronson
   D. Patel
   M. Macias
   D. Reilly

January 1975
64 pp

35c. LITTLE code generator for the UNIVAC 1108 - preliminary remarks.
   S. Gold
   G. Lucans

January 1975
26 pp

35d. LITTLE code generator for the PDP-8.
   R. Rosenthal
   M. Potmesil
   A. Eng
   A. Fogel

January 1975
36 pp

35e. LITTLE code generator for the PDP-11.
   R. Colle
   J. Farrelly
   D. Farkas

January 1975
99 pp

36. Run-time considerations for MIDL.
   E. Deak

November 1974
16 pp

36a. Illustrative examples for MIDL.
   M. Shimasaki

November 1974
17 pp

37. Proposal for MIDL (GLITTLE).
   E. Deak
A library of substantial, important algorithms coded in SETLb exists in a machine readable file. The addition of other algorithms to this library, and the improvement of the documentation and performance of the algorithms that have been established, is an ongoing project. Contents are as follows.

1. deck binres (415 cards)
   coder: e. schonberg
   an automatic theorem prover operating on statements in the sentential calculus. produces shortest proofs via a breadth first tree search when binary resolution is used, or longer proofs in less time when hyper-resolution is employed.

2. deck typevar (543 cards)
   coder: k. abdali
   given the graph of a program and some information about its assignment statements, this algorithm finds the types that a variable can assume during the execution of the program.

3. deck matchup (147 cards)
   coder: g. whitehead
   a modification of marshall hall's algorithm for the marriage problem which will yield a maximal system of
Part 4 - the SFIL algorithms library.

4. deck topdata (112 cards)  
dock topdowna (286 cards)  
dock topdownb (243 cards)  
coder: s. marateck  
        s. brown

5. deck mckeman (531 cards)  
coder: i. kaye

6. deck chomsnf (164 cards)  
coder: n. anthony

7. deck gennapp (470 cards)  
dock gennspd (7 cards)  
coder: y. feinroth

8. deck eulergr (117 cards)  
coder: h. mullish

9. deck lexgena (477 cards)  
dock lexgenb (65 cards)  
dock lexgenc (43 cards)  
coder: t. polacek

10. deck heurmac (130 cards)  
dock heura (202 cards)  
dock heurb (376 cards)  
dock heurc (91 cards)  
dock heurd (92 cards)  
dock heure (82 cards)  
dock heurf (257 cards)  
coder: l. welber

11. deck gps (439 cards)  
coder: a. getzler

distinct representatives.  
a top down parser and its input data,  
complete with a bootstrapping meta-
compiler that operates on an extended  
backus normal form description of the  
language.

generates the mckeman tables (a  
series of generalized precedence  
tables) using backus normal form  
grahm as its input. sample input  
data is included.

removes null variables from a context  
free input grammar (described by its  
productions) and puts it into chomsky  
normal form. sample input data is  
included.

a generalized nodal span parser, with  
attributes. sample input data is  
included.

a setlb coding of l. euler's graph  
tracing algorithm usually associated  
with the bridges of koenigsberg.

the inputs to this mini-system are  
tables describing the character set,  
character types, and actions to be  
taken during the lexical scan of an  
arbitrary language. the output is a  
fortran program which is a working  
lexical scanner for the described  
language, complete with token file  
and error message generators.

six complete independent heuristic  
search procedures, mostly due to  
nilsson, preceded by a deck of macros  
which they all use. the algorithms  
are: a general path finder, a tree  
search, a uniform cost search, a  
breadth first search, a depth first  
search, an and/or tree search for a  
game strategy.

a stripped down version of ernst,  
newell and shaw's general problem...
Part 4 - the SETL algorithms library.

12. deck scgraph (100 cards)
dead partree (128 cards)
dead balance (283 cards)
coder: w. tsui
two separate algorithms to find the strongly connected regions of a directed graph, a program to generate the partitions of a number; a program to generate all binary trees. two separate algorithms for assembly line balancing.

13. deck poly (283 cards)
coder: e. guth
a collection of routines for the standard algebraic manipulations of polynomials. test input is included.

14. deck graal (515 cards)
coder: g. weinberger
a SETL representation of the university of maryland graph manipulation language (graal) of rheinhold, basili and mesztenyi as explained in SETL newsletter 115. sample input data is included.

15. deck heurgra (207 cards)
coder: l. welber
two separate algorithms to find the strongly connected regions of a directed graph, a program to generate the partitions of a number; a program to generate all binary trees. two separate algorithms for assembly line balancing.

Part 5 - the SETL test packages.

***

a library of test programs, coded in SETL, currently resides on a machine readable file. this library is frequently used to spot bugs in compiler modifications, and to provide some standards for timing studies. the algorithms vary widely in size, content, and coding style.

***

1. huffman (75 cards) - produces a huffman tree and table for unique bit string encoding given a set of characters and a frequency of use function over that set.

2. miscperm (50 cards) - contains short programs to make a sequence out of a tuple: compose two functions into one; obtain the inverse of a function; obtain the cycle form of a permutation; obtain the inverse of a permutation; obtain the inverse of a permutation given in cycle form.
3. perm (49 cards) - generates all permutations of n objects in lexicographic order.

4. median (139 cards) - finds the k-th number (in ascending order) of a given set of numbers. This algorithm, due to Floyd, et al in 1971, runs in linear time as a function of the number of items in the given set.

5. pocksor (53 cards) - a radix sort in which the items to be sorted and the radix are input parameters.

6. treeprint (198 cards) - prints binary or ordered trees in a tree-like format.

7. fordj (157 cards) - the ford-johnson tournament sort algorithm (a complicated minimum comparison sort).

8. setup and dsetup (180 cards) - reads SETL code, and prepares a string and some tables for the lexical scanner.

9. intprint (240 cards) - prints the flow-graph of a program in flowchart-like format given a set of paths and a set defining the order in which to print the nodes.

10. twerce (35 cards) - the natural two-way merge for fast in-core sorting.

11. primes (72 cards) - contains short programs to generate primes by the sieve method; generate primes directly from their definition; find the prime factors of a given number.

12. piglatin (27 cards) - string breakup and translation via table lookup or a programmed english-piglatin dictionary.

13. insanity (29 cards) - a backtracking algorithm to solve the instant insanity (colored cubes) puzzle.

14. nodspan (91 cards) - a nodal span parse routine which can apply any production grammar in chomsky normal form to an input string.

15. pascal (16 cards) - a string manipulation and formatting program which prints pascal's triangle neatly.

16. erraut (104 cards) - calculates the structure of the automaton associated with error detection in !lr! parsing.

17. splash (133 cards) - solves all the old bucket problems (e.g. how to get 4 gallons of water given a 3 and a 5 gallon bucket).

18. maxflow (103 cards) - a package to find a path in an ordered graph; determine the maximum flow in a network; and apply the maximum flow algorithm to the matching problem.