CONTENTS

Contents. 1

Acknowledgements. 9

Preface. 10

Survey. 15

Index. 20

Symbols. 29

Functions. 31

References. 36

&1 The standard semantics of Sal. 44

&2 The store semantics of Sal. 56

&3 The stack semantics of Sal. 68

&4 The interpreter for Sal. 80

&5 The translater for Sal. 92

&6 The compiler for Sal. 103

&7 Some syntactic transformations on Sal. 114

&8 Some syntactic constraints on Sal. 123

&9 The auxiliary valuations for Sal. 132

&10 The consecution semantics of Sap. 142

&11 The control semantics of Sap. 148

&12 The consecution semantics of Sam. 154

&13 The pointer semantics of Sam. 159

&14 The loader for Sam. 164

1. Fundamental concepts. 169

- 1.1. Formal descriptions. 169
  - 1.1.1. Some syntactic aspects of valuations. 169
  - 1.1.2. Further features. 172
  - 1.1.3. Lambda notation. 176
- 1.2. Syntactic domains. 179
  - 1.2.1. The purpose of syntactic domains. 179
  - 1.2.2. Constants. 180
  - 1.2.3. Names. 181
  - 1.2.4. Expressions. 182
  - 1.2.5. Abstractions. 184
  - 1.2.6. Sequencers. 186
  - 1.2.7. Declarations. 187
  - 1.2.8. Blocks. 189

1.2.9. Commands. 190

1.3. Textual notions. 191

1.3.1. Scopes. 191

1.3.2. Ranges. 194

1.3.3. Environments. 198

1.4. Temporal notions. 202

1.4.1. Stores. 202

1.4.2. Sharing. 207

1.4.3. Extents. 211

1.4.4. Side effects. 215

1.4.5. Input and output. 218

1.5. Sequencing notions. 219

1.5.1. Answers. 219

1.5.2. Command continuations. 221

1.5.3. Expression continuations. 224

1.5.4. Sorts of sequencer. 228

1.5.5. Labels. 230

1.5.6. Kinds of block. 235

1.5.7. Additional equations. 236

1.5.8. Coroutines. 243

1.5.9. Serial and parallel programs. 245

1.6. Semantic domains. 247

1.6.1. Basic values. 247

1.6.2. Stored and denoted values. 248

1.6.3. Expressed and witnessed values. 249

1.6.4. Abstract data structures. 250

1.6.5. The choice of semantic domains. 253

1.7. Types. 254

1.7.1. Coercions. 254

1.7.2. Modes. 258

1.8. Abstractions. 261

1.8.1. The kinship between abstractions and declarations. 261

1.8.2. Values for abstractions. 265

1.8.3. Fluid abstractions. 267

1.8.4. Delaying the execution of parameters. 272

1.8.5. Coercing the execution of parameters. 276

1.8.6. Frozen abstractions. 280

1.9. Declarations. 283

1.9.1. Declaration continuations. 283

1.9.2. Simple declarations. 285

1.9.3. Multiple declarations. 291

1.9.4. Simple recursive declarations by reference. 296

1.9.5. Simple recursive declarations by incidence. 299

1.9.6. Mutually recursive declarations. 305

1.9.7. Some recursive expressions. 307

2. Mathematical foundations. 309

2.1. The graph model. 309

2.1.1. Enumeration operators. 309

2.2. Spaces. 313

2.2.1. Complete lattices. 313

2.2.2. Flat lattices. 315

2.2.3. Products and sums. 320

2.3. Mappings. 324

2.3.1. Function spaces. 324

2.3.2. Fixed points. 327

2.3.3. Proposition. 330

2.3.4. Proposition. 331

2.3.5. Proposition. 332

2.3.6. Proposition. 333

2.3.7. Proposition. 334

2.3.8. Proposition. 335

2.3.9. Proposition. 336

2.4. Constructing domains. 337

2.4.1. Functors. 337

2.4.2. The retractions defined on a universal domain. 339

2.4.3. Limit spaces. 343

2.4.4. Selection operators. 345

2.4.5. Lambda calculus. 347

2.4.6. Algebraic lattices. 349

2.5. Comparing domains. 353

2.5.1. Predictors. 353

2.5.2. Proposition. 355

2.5.3. Proposition. 357

2.5.4. The diacritical convention. 359

2.5.5. The predicates defined on a universal domain. 361

2.5.6. Proposition. 366

3. Standard semantics. 369

3.1. The general form of the semantic equations. 369

3.1.1. Variations in the presentation. 369

3.1.2. The structure of syntactic domains. 374

3.1.3. The nature of semantic valuations. 376

3.1.4. The need for recursive domain equations. 378

3.2. An informal description of Sal. 382

3.2.1. General design considerations. 382

3.2.2. Constants. 385

3.2.3. Names. 388

3.2.4. Expressions. 389

3.2.5. Abstractions. 391

3.2.6. Sequencers. 392

3.2.7. Declarations. 393

3.2.8. Blocks. 394

3.2.9. Commands. 395

3.3. The operations on the domains. 397

3.3.1. The value domains and the continuation domains. 397

3.3.2. Some basic functions acting on stores. 401

3.3.3. Some basic functions acting on environments. 405

3.3.4. Functions involving continuations. 407

3.3.5. Functors yielding the domains. 409

3.4. The semantic equations for Sal. 413

3.4.1. The valuations $\mathfrak{B}$, $\mathfrak{O}$ and $\mathfrak{W}$. 413

3.4.2. The valuations $\mathfrak{E}$, $\mathfrak{L}$ and $\mathfrak{R}$. 417

3.4.3. The valuations $\mathfrak{F}$ and $\mathfrak{Z}$. 421

3.4.4. The valuation $\mathfrak{D}$. 424

3.4.5. The valuations $\mathfrak{T}$, $\mathfrak{J}$ and $\mathfrak{K}$. 426

3.4.6. The valuation $\mathfrak{G}$. 430

3.4.7. The valuation $\mathfrak{C}$. 432

3.4.8. The valuations $\mathfrak{P}$, $\mathfrak{Q}$, $\mathfrak{J}$ and $\mathfrak{K}$. 434

3.5. Some elementary properties of the semantic equations. 437

3.5.1. Structural induction. 437

3.5.2. Occurrences of names. 441

3.5.3. The pruning of environments. 443

3.5.4. Free variable lists. 445

3.5.5. Proposition. 447

3.5.6. Proposition. 448

3.5.7. Exits from expressions. 449

3.5.8. Syntactic transformations preserving meaning. 450

3.6. Elementary equivalences. 454

3.6.1. Replaceable commands. 454

3.6.2. Proposition. 458

3.6.3. Equivalent blocks. 461

3.6.4. Proposition. 464
3.6.5. Proposition. 465
3.6.6. Proposition. 467
3.6.7. Proposition. 469
3.7. Further standard uses of the semantic equations. 471
3.7.1. Routines. 471
3.7.2. Proposition. 474
3.7.3. Other auxiliary constructions. 475
3.7.4. Proposition. 478
3.7.5. Proposition. 479
3.7.6. An induction rule for loops. 480
3.7.7. Proposition. 482
3.7.8. Assertions about commands. 485
3.7.9. Predicate transformers. 491
3.8. Restricted equivalences. 501
3.8.1. Possible forms of constraint. 501
3.8.2. Similarity. 504
3.8.3. Some useful properties of least fixed points. 509
3.8.4. Illustrative examples. 514
3.8.5. Commutativity. 520
3.8.6. Proposition. 527
3.8.7. Idempotence. 532
3.8.8. Proposition. 536
3.9. Recursive procedures. 541
3.9.1. Conjugate valuations. 541
3.9.2. Programs without jumps. 544
3.9.3. Proposition. 550
3.9.4. Proposition. 551
3.9.5. Simple applications. 558
3.9.6. Some induction rules about recursion. 561
3.9.7. Alterations to programs. 567
3.9.8. Termination. 569
4. Store semantics. 574
4.1. State vectors. 574
4.1.1. Some nasty equivalence problems. 574
4.1.2. Stacks. 577
4.1.3. The recurrence of states. 581
4.1.4. The conservation of environments. 585
4.1.5. The formation of the domains. 588
4.1.6. The vectorial convention. 591

4.2. Recursive relations. 592

4.2.1. Congruent equations. 592

4.2.2. The links required by the domains. 595

4.2.3. The forging of these links. 598

4.2.4. Subsidiary predicates. 606

4.2.5. Syntactic predicates. 607

4.2.6. Storage allocation. 611

4.2.7. The removal of continuations. 614

4.3. The congruence between standard semantics and store semantics. 623

4.3.1. Lemma. 623

4.3.2. Lemma. 625

4.3.3. Lemma. 626

4.3.4. Lemma. 627

4.3.5. Lemma. 629

4.3.6. Lemma. 632

4.3.7. Theorem. 634

4.4. Interpreters. 635

4.4.1. Controls. 635

4.4.2. Orders. 639

4.4.3. An assortment of state valuations. 643

4.4.4. The folding of environments. 647

4.5. The identity between store semantics and consecution semantics. 651

4.5.1. Consecutions. 651

4.5.2. Proposition. 656

4.5.3. Proposition. 657

4.5.4. Proposition. 658

4.5.5. Proposition. 659

4.6. The conformity between consecution semantics and control semantics. 663

4.6.1. Computational induction. 663

4.6.2. The explicit connection between functions and program texts. 668

4.6.3. Lemma. 674

4.6.4. Proposition. 681

4.6.5. The implicit connection between functions and program texts. 683

4.6.6. Lemma. 691

4.6.7. Proposition. 696

4.6.8. Theorem. 699

4.6.9. The assumptions underlying the basic result. 701

5. Stack semantics. 708

5.1. Idealized versions of realistic implementations. 708

5.1.1. Inefficiencies in general techniques. 708

5.1.2. Displays. 711

5.1.3. Label and procedure values. 715

5.1.4. The restoration of stores. 719

5.1.5. Chains. 722

5.1.6. The creation of environments. 725

5.1.7. Alternative approaches. 729

5.1.8. Syntactic constraints ensuring validity. 732

5.1.9. Some semantic implications of these constraints. 736

5.2. Preparations for an inductive proof. 739

5.2.1. Shortcomings of direct connections. 739

5.2.2. Tracing algorithms. 743

5.2.3. Resemblances between states. 747

5.2.4. Correspondences between values. 750

5.2.5. Predictors providing suitable properties. 756

5.2.6. The syntactic counterparts of these properties. 763

5.2.7. Proposition. 768

5.2.8. Proposition. 770

5.2.9. Proposition. 772

5.3. The congruence between store semantics and stack semantics. 773

5.3.1. Lemma. 773

5.3.2. Lemma. 775

5.3.3. Lemma. 777

5.3.4. Lemma. 779

5.3.5. Lemma. 781

5.3.6. Lemma. 783

5.3.7. Lemma. 785

5.3.8. Lemma. 788

5.3.9. Theorem. 791

5.4. Compilers. 794

5.4.1. Code generation. 794

5.4.2. Instructions. 798

5.4.3. Code relocation. 801

5.4.4. Proposition. 805

5.4.5. Proposition. 807

5.4.6. The decomposition of semantic equations. 808

5.5. The identity between stack semantics and consecution semantics. 811

5.5.1. Lemma. 811

5.5.2. Lemma. 813

5.5.3. Lemma. 815

5.5.4. Lemma. 817

5.5.5. Lemma. 821

5.5.6. Lemma. 823

5.5.7. Theorem. 827

5.6. The conformity between consecution semantics and pointer semantics. 828

5.6.1. Pointers. 828

5.6.2. The explicit connection between functions and bit patterns. 832

5.6.3. Lemma. 837

5.6.4. Proposition. 838

5.6.5. The implicit connection between functions and bit patterns. 840

5.6.6. Lemma. 845

5.6.7. Proposition. 847

5.6.8. Theorem. 849

5.6.9. The correctness of implementations. 851