

Tabla de contenido

| | |
|--|----------|
| Prefacio | v |
| 1 Introducción | 1 |
| 1.1 Procesadores de lenguaje | 1 |
| 1.1.1 Ejercicios para la sección 1.1. | 3 |
| 1.2 La estructura de un compilador | 4 |
| 1.2.1 Análisis de léxico | 5 |
| 1.2.2 Análisis sintáctico | 8 |
| 1.2.3 Análisis semántico | 8 |
| 1.2.4 Generación de código intermedio. | 9 |
| 1.2.5 Optimización de código | 10 |
| 1.2.6 Generación de código | 10 |
| 1.2.7 Administración de la tabla de símbolos | 11 |
| 1.2.8 El agrupamiento de fases en pasadas | 11 |
| 1.2.9 Herramientas de construcción de compiladores | 12 |
| 1.3 La evolución de los lenguajes de programación | 12 |
| 1.3.1 El avance a los lenguajes de alto nivel | 13 |
| 1.3.2 Impactos en el compilador | 14 |
| 1.3.3 Ejercicios para la sección 1.3. | 14 |
| 1.4 La ciencia de construir un compilador | 15 |
| 1.4.1 Modelado en el diseño e implementación de compiladores | 15 |
| 1.4.2 La ciencia de la optimización de código | 15 |
| 1.5 Aplicaciones de la tecnología de compiladores | 17 |
| 1.5.1 Implementación de lenguajes de programación de alto nivel. | 17 |
| 1.5.2 Optimizaciones para las arquitecturas de computadoras | 19 |
| 1.5.3 Diseño de nuevas arquitecturas de computadoras | 21 |
| 1.5.4 Traducciones de programas. | 22 |
| 1.5.5 Herramientas de productividad de software | 23 |
| 1.6 Fundamentos de los lenguajes de programación | 25 |
| 1.6.1 La distinción entre estático y dinámico | 25 |
| 1.6.2 Entornos y estados | 26 |
| 1.6.3 Alcance estático y estructura de bloques | 28 |
| 1.6.4 Control de acceso explícito | 31 |
| 1.6.5 Alcance dinámico. | 31 |
| 1.6.6 Mecanismos para el paso de parámetros. | 33 |

| | | |
|----------|--|-----------|
| 1.6.7 | Uso de alias | 35 |
| 1.6.8 | Ejercicios para la sección 1.6. | 35 |
| 1.7 | Resumen del capítulo 1 | 36 |
| 1.8 | Referencias para el capítulo 1 | 38 |
| 2 | Un traductor simple orientado a la sintaxis | 39 |
| 2.1 | Introducción | 40 |
| 2.2 | Definición de sintaxis. | 42 |
| 2.2.1 | Definición de gramáticas | 42 |
| 2.2.2 | Derivaciones | 44 |
| 2.2.3 | Árboles de análisis sintáctico | 45 |
| 2.2.4 | Ambigüedad | 47 |
| 2.2.5 | Asociatividad de los operadores | 48 |
| 2.2.6 | Precedencia de operadores | 48 |
| 2.2.7 | Ejercicios para la sección 2.2. | 51 |
| 2.3 | Traducción orientada a la sintaxis. | 52 |
| 2.3.1 | Notación postfija. | 53 |
| 2.3.2 | Atributos sintetizados | 54 |
| 2.3.3 | Definiciones simples orientadas a la sintaxis | 56 |
| 2.3.4 | Recorridos de los árboles | 56 |
| 2.3.5 | Esquemas de traducción. | 57 |
| 2.3.6 | Ejercicios para la sección 2.3. | 60 |
| 2.4 | Análisis sintáctico | 60 |
| 2.4.1 | Análisis sintáctico tipo arriba-abajo | 61 |
| 2.4.2 | Análisis sintáctico predictivo. | 64 |
| 2.4.3 | Cuándo usar las producciones ϵ | 65 |
| 2.4.4 | Diseño de un analizador sintáctico predictivo | 66 |
| 2.4.5 | Recursividad a la izquierda. | 67 |
| 2.4.6 | Ejercicios para la sección 2.4. | 68 |
| 2.5 | Un traductor para las expresiones simples | 68 |
| 2.5.1 | Sintaxis abstracta y concreta | 69 |
| 2.5.2 | Adaptación del esquema de traducción | 70 |
| 2.5.3 | Procedimientos para los no terminales. | 72 |
| 2.5.4 | Simplificación del traductor | 73 |
| 2.5.5 | El programa completo | 74 |
| 2.6 | Análisis léxico. | 76 |
| 2.6.1 | Eliminación de espacio en blanco y comentarios | 77 |
| 2.6.2 | Lectura adelantada | 78 |
| 2.6.3 | Constantes | 78 |
| 2.6.4 | Reconocimiento de palabras clave e identificadores | 79 |
| 2.6.5 | Un analizador léxico | 81 |
| 2.6.6 | Ejercicios para la sección 2.6. | 84 |
| 2.7 | Tablas de símbolos | 85 |
| 2.7.1 | Tabla de símbolos por alcance. | 86 |
| 2.7.2 | El uso de las tablas de símbolos | 89 |

| | | |
|----------|--|------------|
| 2.8 | Generación de código intermedio. | 91 |
| 2.8.1 | Dos tipos de representaciones intermedias | 91 |
| 2.8.2 | Construcción de árboles sintácticos | 92 |
| 2.8.3 | Comprobación estática. | 97 |
| 2.8.4 | Código de tres direcciones | 99 |
| 2.8.5 | Ejercicios para la sección 2.8. | 105 |
| 2.9 | Resumen del capítulo 2 | 105 |
| 3 | Análisis léxico | 109 |
| 3.1 | La función del analizador léxico | 109 |
| 3.1.1 | Comparación entre análisis léxico y análisis sintáctico. | 110 |
| 3.1.2 | Tokens, patrones y lexemas | 111 |
| 3.1.3 | Atributos para los tokens | 112 |
| 3.1.4 | Errores léxicos. | 113 |
| 3.1.5 | Ejercicios para la sección 3.1. | 114 |
| 3.2 | Uso de búfer en la entrada | 115 |
| 3.2.1 | Pares de búferes | 115 |
| 3.2.2 | Centinelas. | 116 |
| 3.3 | Especificación de los tokens | 116 |
| 3.3.1 | Cadenas y lenguajes. | 117 |
| 3.3.2 | Operaciones en los lenguajes. | 119 |
| 3.3.3 | Expresiones regulares. | 120 |
| 3.3.4 | Definiciones regulares. | 123 |
| 3.3.5 | Extensiones de las expresiones regulares | 124 |
| 3.3.6 | Ejercicios para la sección 3.3. | 125 |
| 3.4 | Reconocimiento de tokens | 128 |
| 3.4.1 | Diagramas de transición de estados | 130 |
| 3.4.2 | Reconocimiento de las palabras reservadas y los identificadores | 132 |
| 3.4.3 | Finalización del bosquejo | 133 |
| 3.4.4 | Arquitectura de un analizador léxico basado en diagramas de transición de estados | 134 |
| 3.4.5 | Ejercicios para la sección 3.4. | 136 |
| 3.5 | El generador de analizadores léxicos Lex | 140 |
| 3.5.1 | Uso de Lex | 140 |
| 3.5.2 | Estructura de los programas en Lex | 141 |
| 3.5.3 | Resolución de conflictos en Lex. | 144 |
| 3.5.4 | El operador adelantado | 144 |
| 3.5.5 | Ejercicios para la sección 3.5. | 146 |
| 3.6 | Autómatas finitos | 147 |
| 3.6.1 | Autómatas finitos no deterministas | 147 |
| 3.6.2 | Tablas de transición. | 148 |
| 3.6.3 | Aceptación de las cadenas de entrada mediante los autómatas | 149 |
| 3.6.4 | Autómatas finitos deterministas | 149 |
| 3.6.5 | Ejercicios para la sección 3.6. | 151 |

| | | |
|----------|--|------------|
| 3.7 | De las expresiones regulares a los autómatas | 152 |
| 3.7.1 | Conversión de un AFN a AFD | 152 |
| 3.7.2 | Simulación de un AFN | 156 |
| 3.7.3 | Eficiencia de la simulación de un AFN | 157 |
| 3.7.4 | Construcción de un AFN a partir de una expresión regular | 159 |
| 3.7.5 | Eficiencia de los algoritmos de procesamiento de cadenas | 163 |
| 3.7.6 | Ejercicios para la sección 3.7. | 166 |
| 3.8 | Diseño de un generador de analizadores léxicos | 166 |
| 3.8.1 | La estructura del analizador generado | 167 |
| 3.8.2 | Coincidencia de patrones con base en los AFNs | 168 |
| 3.8.3 | AFDs para analizadores léxicos | 170 |
| 3.8.4 | Implementación del operador de preanálisis | 171 |
| 3.8.5 | Ejercicios para la sección 3.8. | 172 |
| 3.9 | Optimización de los buscadores por concordancia de patrones basados en AFD. | 173 |
| 3.9.1 | Estados significativos de un AFN | 173 |
| 3.9.2 | Funciones calculadas a partir del árbol sintáctico | 175 |
| 3.9.3 | Cálculo de <i>anulable</i> , <i>primerapos</i> y <i>ultimapos</i> | 176 |
| 3.9.4 | Cálculo de <i>siguientepos</i> | 177 |
| 3.9.5 | Conversión directa de una expresión regular a un AFD | 179 |
| 3.9.6 | Minimización del número de estados de un AFD | 180 |
| 3.9.7 | Minimización de estados en los analizadores léxicos | 184 |
| 3.9.8 | Intercambio de tiempo por espacio en la simulación de un AFD. | 185 |
| 3.9.9 | Ejercicios para la sección 3.9. | 186 |
| 3.10 | Resumen del capítulo 3 | 187 |
| 3.11 | Referencias para el capítulo 3. | 189 |
| 4 | Análisis sintáctico | 191 |
| 4.1 | Introducción | 192 |
| 4.1.1 | La función del analizador sintáctico. | 192 |
| 4.1.2 | Representación de gramáticas | 193 |
| 4.1.3 | Manejo de los errores sintácticos | 194 |
| 4.1.4 | Estrategias para recuperarse de los errores. | 195 |
| 4.2 | Gramáticas libres de contexto. | 197 |
| 4.2.1 | La definición formal de una gramática libre de contexto | 197 |
| 4.2.2 | Convenciones de notación. | 198 |
| 4.2.3 | Derivaciones | 199 |
| 4.2.4 | Árboles de análisis sintáctico y derivaciones | 201 |
| 4.2.5 | Ambigüedad | 203 |
| 4.2.6 | Verificación del lenguaje generado por una gramática | 204 |
| 4.2.7 | Comparación entre gramáticas libres de contexto y expresiones regulares | 205 |
| 4.2.8 | Ejercicios para la sección 4.2. | 206 |
| 4.3 | Escritura de una gramática | 209 |
| 4.3.1 | Comparación entre análisis léxico y análisis sintáctico. | 209 |
| 4.3.2 | Eliminación de la ambigüedad | 210 |

| | | |
|-------|---|-----|
| 4.3.3 | Eliminación de la recursividad por la izquierda. | 212 |
| 4.3.4 | Factorización por la izquierda | 214 |
| 4.3.5 | Construcciones de lenguajes que no son libres de contexto. | 215 |
| 4.3.6 | Ejercicios para la sección 4.3. | 216 |
| 4.4 | Análisis sintáctico descendente | 217 |
| 4.4.1 | Análisis sintáctico de descenso recursivo | 219 |
| 4.4.2 | PRIMERO y SIGUIENTE | 220 |
| 4.4.3 | Gramáticas LL(1) | 222 |
| 4.4.4 | Análisis sintáctico predictivo no recursivo | 226 |
| 4.4.5 | Recuperación de errores en el análisis sintáctico predictivo | 228 |
| 4.4.6 | Ejercicios para la sección 4.4. | 231 |
| 4.5 | Análisis sintáctico ascendente | 233 |
| 4.5.1 | Reducciones | 234 |
| 4.5.2 | Poda de mangos | 235 |
| 4.5.3 | Análisis sintáctico de desplazamiento-reducción | 236 |
| 4.5.4 | Conflictos durante el análisis sintáctico de desplazamiento-reducción | 238 |
| 4.5.5 | Ejercicios para la sección 4.5. | 240 |
| 4.6 | Introducción al análisis sintáctico LR: SLR (LR simple) | 241 |
| 4.6.1 | ¿Por qué analizadores sintácticos LR? | 241 |
| 4.6.2 | Los elementos y el autómata LR(0). | 242 |
| 4.6.3 | El algoritmo de análisis sintáctico LR | 248 |
| 4.6.4 | Construcción de tablas de análisis sintáctico SLR. | 252 |
| 4.6.5 | Prefijos viables | 256 |
| 4.6.6 | Ejercicios para la sección 4.6. | 257 |
| 4.7 | Analizadores sintácticos LR más poderosos | 259 |
| 4.7.1 | Elementos LR(1) canónicos. | 260 |
| 4.7.2 | Construcción de conjuntos de elementos LR(1). | 261 |
| 4.7.3 | Tablas de análisis sintáctico LR(1) canónico. | 265 |
| 4.7.4 | Construcción de tablas de análisis sintáctico LALR | 266 |
| 4.7.5 | Construcción eficiente de tablas de análisis sintáctico LALR | 270 |
| 4.7.6 | Compactación de las tablas de análisis sintáctico LR | 275 |
| 4.7.7 | Ejercicios para la sección 4.7. | 277 |
| 4.8 | Uso de gramáticas ambiguas. | 278 |
| 4.8.1 | Precedencia y asociatividad para resolver conflictos | 279 |
| 4.8.2 | La ambigüedad del “else colgante” | 281 |
| 4.8.3 | Recuperación de errores en el análisis sintáctico LR | 283 |
| 4.8.4 | Ejercicios para la sección 4.8. | 285 |
| 4.9 | Generadores de analizadores sintácticos. | 287 |
| 4.9.1 | El generador de analizadores sintácticos Yacc. | 287 |
| 4.9.2 | Uso de Yacc con gramáticas ambiguas. | 291 |
| 4.9.3 | Creación de analizadores léxicos de Yacc con Lex | 294 |
| 4.9.4 | Recuperación de errores en Yacc. | 295 |
| 4.9.5 | Ejercicios para la sección 4.9. | 297 |
| 4.10 | Resumen del capítulo 4 | 297 |
| 4.11 | Referencias para el capítulo 4. | 300 |

| | | |
|----------|---|------------|
| 5 | Traducción orientada por la sintaxis | 303 |
| 5.1 | Definiciones dirigidas por la sintaxis | 304 |
| 5.1.1 | Atributos heredados y sintetizados | 304 |
| 5.1.2 | Evaluación de una definición dirigida por la sintaxis en los nodos de un árbol de análisis sintáctico | 306 |
| 5.1.3 | Ejercicios para la sección 5.1. | 309 |
| 5.2 | Órdenes de evaluación para las definiciones dirigidas por la sintaxis | 310 |
| 5.2.1 | Gráficos de dependencias | 310 |
| 5.2.2 | Orden de evaluación | 312 |
| 5.2.3 | Definiciones con atributos sintetizados. | 312 |
| 5.2.4 | Definiciones con atributos heredados. | 313 |
| 5.2.5 | Reglas semánticas con efectos adicionales controlados | 314 |
| 5.2.6 | Ejercicios para la sección 5.2. | 317 |
| 5.3 | Aplicaciones de la traducción orientada por la sintaxis | 318 |
| 5.3.1 | Construcción de árboles de análisis sintáctico. | 318 |
| 5.3.2 | La estructura de tipos | 321 |
| 5.3.3 | Ejercicios para la sección 5.3. | 323 |
| 5.4 | Esquemas de traducción orientados por la sintaxis | 323 |
| 5.4.1 | Esquemas de traducción postfijos | 324 |
| 5.4.2 | Implementación de esquemas de traducción orientados a la sintaxis postfijo con la pila del analizador sintáctico | 325 |
| 5.4.3 | Esquemas de traducción orientados a la sintaxis con acciones dentro de las producciones | 327 |
| 5.4.4 | Eliminación de la recursividad por la izquierda de los esquemas de traducción orientados a la sintaxis | 328 |
| 5.4.5 | Esquemas de traducción orientados a la sintaxis para definiciones con atributos heredados por la izquierda | 331 |
| 5.4.6 | Ejercicios para la sección 5.4. | 336 |
| 5.5 | Implementación de definiciones dirigidas por la sintaxis con atributos heredados por la izquierda | 337 |
| 5.5.1 | Traducción durante el análisis sintáctico de descenso recursivo. | 338 |
| 5.5.2 | Generación de código al instante. | 340 |
| 5.5.3 | Las definiciones dirigidas por la sintaxis con atributos heredados por la izquierda y el análisis sintáctico LL | 343 |
| 5.5.4 | Análisis sintáctico ascendente de las definiciones dirigidas por la sintaxis con atributos heredados por la izquierda. | 348 |
| 5.5.5 | Ejercicios para la sección 5.5. | 352 |
| 5.6 | Resumen del capítulo 5 | 353 |
| 5.7 | Referencias para el capítulo 5. | 354 |
| 6 | Generación de código intermedio | 357 |
| 6.1 | Variantes de los árboles sintácticos | 358 |
| 6.1.1 | Grafo dirigido acíclico para las expresiones | 359 |
| 6.1.2 | El método número de valor para construir GDAs | 360 |
| 6.1.3 | Ejercicios para la sección 6.1. | 362 |

| | | |
|-------|---|-----|
| 6.2 | Código de tres direcciones | 363 |
| 6.2.1 | Direcciones e instrucciones | 364 |
| 6.2.2 | Cuádruplos | 366 |
| 6.2.3 | Tripletas | 367 |
| 6.2.4 | Forma de asignación individual estática | 369 |
| 6.2.5 | Ejercicios para la sección 6.2. | 370 |
| 6.3 | Tipos y declaraciones. | 370 |
| 6.3.1 | Expresiones de tipos | 371 |
| 6.3.2 | Equivalencia de tipos | 372 |
| 6.3.3 | Declaraciones | 373 |
| 6.3.4 | Distribución del almacenamiento para los nombres locales. | 373 |
| 6.3.5 | Secuencias de las declaraciones | 376 |
| 6.3.6 | Campos en registros y clases. | 376 |
| 6.3.7 | Ejercicios para la sección 6.3. | 378 |
| 6.4 | Traducción de expresiones | 378 |
| 6.4.1 | Operaciones dentro de expresiones | 378 |
| 6.4.2 | Traducción incremental | 380 |
| 6.4.3 | Direccionamiento de los elementos de un arreglo. | 381 |
| 6.4.4 | Traducción de referencias a arreglos | 383 |
| 6.4.5 | Ejercicios para la sección 6.4. | 384 |
| 6.5 | Comprobación de tipos | 386 |
| 6.5.1 | Reglas para la comprobación de tipos | 387 |
| 6.5.2 | Conversiones de tipos. | 388 |
| 6.5.3 | Sobrecarga de funciones y operadores | 390 |
| 6.5.4 | Inferencia de tipos y funciones polimórficas | 391 |
| 6.5.5 | Un algoritmo para la unificación | 395 |
| 6.5.6 | Ejercicios para la sección 6.5. | 398 |
| 6.6 | Flujo de control. | 399 |
| 6.6.1 | Expresiones booleanas | 399 |
| 6.6.2 | Código de corto circuito. | 400 |
| 6.6.3 | Instrucciones de flujo de control | 401 |
| 6.6.4 | Traducción del flujo de control de las expresiones booleanas | 403 |
| 6.6.5 | Evitar gotos redundantes | 405 |
| 6.6.6 | Valores booleanos y código de salto. | 408 |
| 6.6.7 | Ejercicios para la sección 6.6. | 408 |
| 6.7 | Parcheo de retroceso (backpatch) | 410 |
| 6.7.1 | Generación de código de una pasada, mediante parcheo de retroceso | 410 |
| 6.7.2 | Técnica de parcheo de retroceso para las expresiones booleanas | 411 |
| 6.7.3 | Instrucciones de flujo de control | 413 |
| 6.7.4 | Instrucciones break, continue y goto | 416 |
| 6.7.5 | Ejercicios para la sección 6.7. | 417 |
| 6.8 | Instrucciones switch. | 418 |
| 6.8.1 | Traducción de las instrucciones switch. | 419 |
| 6.8.2 | Traducción orientada por la sintaxis de las instrucciones switch | 420 |
| 6.8.3 | Ejercicios para la sección 6.8. | 421 |

| | | |
|----------|--|------------|
| 6.9 | Código intermedio para procedimientos | 422 |
| 6.10 | Resumen del capítulo 6 | 424 |
| 6.11 | Referencias para el capítulo 6 | 425 |
| 7 | Entornos en tiempo de ejecución | 427 |
| 7.1 | Organización del almacenamiento | 427 |
| 7.1.1 | Asignación de almacenamiento estática y dinámica | 429 |
| 7.2 | Asignación de espacio en la pila | 430 |
| 7.2.1 | Árboles de activación | 430 |
| 7.2.2 | Registros de activación | 433 |
| 7.2.3 | Secuencias de llamadas | 436 |
| 7.2.4 | Datos de longitud variable en la pila | 438 |
| 7.2.5 | Ejercicios para la sección 7.2 | 440 |
| 7.3 | Acceso a los datos no locales en la pila | 441 |
| 7.3.1 | Acceso a los datos sin procedimientos anidados | 442 |
| 7.3.2 | Problemas con los procedimientos anidados | 442 |
| 7.3.3 | Un lenguaje con declaraciones de procedimientos anidados | 443 |
| 7.3.4 | Profundidad de anidamiento | 443 |
| 7.3.5 | Enlace de acceso | 445 |
| 7.3.6 | Manipulación de los enlaces de acceso | 447 |
| 7.3.7 | Enlaces de acceso para los parámetros de procedimientos | 448 |
| 7.3.8 | Estructura de datos Display | 449 |
| 7.3.9 | Ejercicios para la sección 7.3 | 451 |
| 7.4 | Administración del montículo | 452 |
| 7.4.1 | El administrador de memoria | 453 |
| 7.4.2 | La jerarquía de memoria de una computadora | 454 |
| 7.4.3 | Localidad en los programas | 455 |
| 7.4.4 | Reducción de la fragmentación | 457 |
| 7.4.5 | Solicitudes de desasignación manual | 460 |
| 7.4.6 | Ejercicios para la sección 7.4 | 463 |
| 7.5 | Introducción a la recolección de basura | 463 |
| 7.5.1 | Metas de diseño para los recolectores de basura | 464 |
| 7.5.2 | Capacidad de alcance | 466 |
| 7.5.3 | Recolectores de basura con conteo de referencias | 468 |
| 7.5.4 | Ejercicios para la sección 7.5 | 470 |
| 7.6 | Introducción a la recolección basada en el rastreo | 470 |
| 7.6.1 | Un recolector básico “marcar y limpiar” | 471 |
| 7.6.2 | Abstracción básica | 473 |
| 7.6.3 | Optimización de “marcar y limpiar” | 475 |
| 7.6.4 | Recolectores de basura “marcar y compactar” | 476 |
| 7.6.5 | Recolectores de copia | 478 |
| 7.6.6 | Comparación de costos | 482 |
| 7.6.7 | Ejercicios para la sección 7.6 | 482 |
| 7.7 | Recolección de basura de pausa corta | 483 |
| 7.7.1 | Recolección de basura incremental | 483 |

| | | |
|----------|--|------------|
| 7.7.2 | Análisis de capacidad alcance incremental | 485 |
| 7.7.3 | Fundamentos de la recolección parcial | 487 |
| 7.7.4 | Recolección de basura generacional | 488 |
| 7.7.5 | El algoritmo del tren | 490 |
| 7.7.6 | Ejercicios para la sección 7.7. | 493 |
| 7.8 | Temas avanzados sobre la recolección de basura. | 494 |
| 7.8.1 | Recolección de basura paralela y concurrente. | 495 |
| 7.8.2 | Reubicación parcial de objetos | 497 |
| 7.8.3 | Recolección conservadora para lenguajes inseguros | 498 |
| 7.8.4 | Referencias débiles. | 498 |
| 7.8.5 | Ejercicios para la sección 7.8. | 499 |
| 7.9 | Resumen del capítulo 7 | 500 |
| 7.10 | Referencias para el capítulo 7. | 502 |
| 8 | Generación de código | 505 |
| 8.1 | Cuestiones sobre el diseño de un generador de código | 506 |
| 8.1.1 | Entrada del generador de código | 507 |
| 8.1.2 | El programa destino. | 507 |
| 8.1.3 | Selección de instrucciones. | 508 |
| 8.1.4 | Asignación de registros. | 510 |
| 8.1.5 | Orden de evaluación | 511 |
| 8.2 | El lenguaje destino | 512 |
| 8.2.1 | Un modelo simple de máquina destino. | 512 |
| 8.2.2 | Costos del programa y las instrucciones. | 515 |
| 8.2.3 | Ejercicios para la sección 8.2. | 516 |
| 8.3 | Direcciones en el código destino | 518 |
| 8.3.1 | Asignación estática | 518 |
| 8.3.2 | Asignación de pila | 520 |
| 8.3.3 | Direcciones para los nombres en tiempo de ejecución | 522 |
| 8.3.4 | Ejercicios para la sección 8.3. | 524 |
| 8.4 | Bloques básicos y grafos de flujo. | 525 |
| 8.4.1 | Bloques básicos | 526 |
| 8.4.2 | Información de siguiente uso. | 528 |
| 8.4.3 | Grafos de flujo | 529 |
| 8.4.4 | Representación de los grafos de flujo | 530 |
| 8.4.5 | Ciclos. | 531 |
| 8.4.6 | Ejercicios para la sección 8.4. | 531 |
| 8.5 | Optimización de los bloques básicos | 533 |
| 8.5.1 | La representación en GDA de los bloques básicos | 533 |
| 8.5.2 | Búsqueda de subexpresiones locales comunes | 534 |
| 8.5.3 | Eliminación del código muerto | 535 |
| 8.5.4 | El uso de identidades algebraicas | 536 |
| 8.5.5 | Representación de referencias a arreglos. | 537 |
| 8.5.6 | Asignaciones de apuntadores y llamadas a procedimientos. | 539 |
| 8.5.7 | Reensamblado de los bloques básicos a partir de los GDAs | 539 |
| 8.5.8 | Ejercicios para la sección 8.5. | 541 |

| | | |
|----------|---|------------|
| 8.6 | Un generador de código simple | 542 |
| 8.6.1 | Descriptores de registros y direcciones | 543 |
| 8.6.2 | El algoritmo de generación de código | 544 |
| 8.6.3 | Diseño de la función <i>obtenReg</i> | 547 |
| 8.6.4 | Ejercicios para la sección 8.6. | 548 |
| 8.7 | Optimización de mirilla (peephole) | 549 |
| 8.7.1 | Eliminación de instrucciones redundantes de carga y almacenamiento | 550 |
| 8.7.2 | Eliminación de código inalcanzable | 550 |
| 8.7.3 | Optimizaciones del flujo de control | 551 |
| 8.7.4 | Simplificación algebraica y reducción por fuerza | 552 |
| 8.7.5 | Uso de características específicas de máquina | 552 |
| 8.7.6 | Ejercicios para la sección 8.7. | 553 |
| 8.8 | Repartición y asignación de registros. | 553 |
| 8.8.1 | Repartición global de registros | 553 |
| 8.8.2 | Conteos de uso | 554 |
| 8.8.3 | Asignación de registros para ciclos externos | 556 |
| 8.8.4 | Asignación de registros mediante la coloración de grafos | 556 |
| 8.8.5 | Ejercicios para la sección 8.8. | 557 |
| 8.9 | Selección de instrucciones mediante la rescritura de árboles. | 558 |
| 8.9.1 | Esquemas de traducción de árboles | 558 |
| 8.9.2 | Generación de código mediante el revestimiento de un árbol de entrada. | 560 |
| 8.9.3 | Coincidencias de los patrones mediante el análisis sintáctico | 563 |
| 8.9.4 | Rutinas para la comprobación semántica | 565 |
| 8.9.5 | Proceso general para igualar árboles | 565 |
| 8.9.6 | Ejercicios para la sección 8.9. | 567 |
| 8.10 | Generación de código óptimo para las expresiones | 567 |
| 8.10.1 | Números de Ershov | 567 |
| 8.10.2 | Generación de código a partir de árboles de expresión etiquetados | 568 |
| 8.10.3 | Evaluación de expresiones con un suministro insuficiente de registros. | 570 |
| 8.10.4 | Ejercicios para la sección 8.10. | 572 |
| 8.11 | Generación de código de programación dinámica | 573 |
| 8.11.1 | Evaluación contigua. | 574 |
| 8.11.2 | El algoritmo de programación dinámica. | 575 |
| 8.11.3 | Ejercicios para la sección 8.11. | 577 |
| 8.12 | Resumen del capítulo 8 | 578 |
| 8.13 | Referencias para el capítulo 8 | 579 |
| 9 | Optimizaciones independientes de la máquina | 583 |
| 9.1 | Las fuentes principales de optimización | 584 |
| 9.1.1 | Causas de redundancia. | 584 |
| 9.1.2 | Un ejemplo abierto: Quicksort | 585 |
| 9.1.3 | Transformaciones que preservan la semántica. | 586 |
| 9.1.4 | Subexpresiones comunes globales | 588 |
| 9.1.5 | Propagación de copias | 590 |
| 9.1.6 | Eliminación de código muerto. | 591 |

| | | |
|-------|---|-----|
| 9.1.7 | Movimiento de código | 592 |
| 9.1.8 | VARIABLES DE INDUCCIÓN Y REDUCCIÓN EN FUERZA | 592 |
| 9.1.9 | Ejercicios para la sección 9.1. | 596 |
| 9.2 | Introducción al análisis del flujo de datos | 597 |
| 9.2.1 | La abstracción del flujo de datos. | 597 |
| 9.2.2 | El esquema del análisis del flujo de datos. | 599 |
| 9.2.3 | Esquemas del flujo de datos en bloques básicos | 600 |
| 9.2.4 | Definiciones de alcance. | 601 |
| 9.2.5 | Análisis de variables vivas | 608 |
| 9.2.6 | Expresiones disponibles | 610 |
| 9.2.7 | Resumen. | 614 |
| 9.2.8 | Ejercicios para la sección 9.2. | 615 |
| 9.3 | Fundamentos del análisis del flujo de datos | 618 |
| 9.3.1 | Semi-lattices | 618 |
| 9.3.2 | Funciones de transferencia | 623 |
| 9.3.3 | El algoritmo iterativo para los marcos de trabajo generales. | 626 |
| 9.3.4 | Significado de una solución de un flujo de datos | 628 |
| 9.3.5 | Ejercicios para la sección 9.3. | 631 |
| 9.4 | Propagación de constantes | 632 |
| 9.4.1 | Valores del flujo de datos para el marco de trabajo de propagación de constantes. | 633 |
| 9.4.2 | La reunión para el marco de trabajo de propagación de constantes | 633 |
| 9.4.3 | Funciones de transferencia para el marco de trabajo de propagación de constantes. | 634 |
| 9.4.4 | Monotonía en el marco de trabajo de propagación de constantes | 635 |
| 9.4.5 | La distributividad del marco de trabajo de propagación de constantes | 635 |
| 9.4.6 | Interpretación de los resultados. | 637 |
| 9.4.7 | Ejercicios para la sección 9.4. | 637 |
| 9.5 | Eliminación de redundancia parcial. | 639 |
| 9.5.1 | Los orígenes de la redundancia | 639 |
| 9.5.2 | ¿Puede eliminarse toda la redundancia? | 642 |
| 9.5.3 | El problema del movimiento de código diferido. | 644 |
| 9.5.4 | Anticipación de las expresiones. | 645 |
| 9.5.5 | El algoritmo de movimiento de código diferido. | 646 |
| 9.5.6 | Ejercicios para la sección 9.5. | 655 |
| 9.6 | Ciclos en los grafos de flujo | 655 |
| 9.6.1 | Dominadores. | 656 |
| 9.6.2 | Ordenamiento “primero en profundidad” | 660 |
| 9.6.3 | Aristas en un árbol de expansión con búsqueda en profundidad | 661 |
| 9.6.4 | Aristas posteriores y capacidad de reducción | 662 |
| 9.6.5 | Profundidad de un grafo de flujo. | 665 |
| 9.6.6 | Ciclos naturales. | 665 |
| 9.6.7 | Velocidad de convergencia de los algoritmos de flujos de datos iterativos. | 667 |
| 9.6.8 | Ejercicios para la sección 9.6. | 669 |

| | | |
|-----------|---|------------|
| 9.7 | Análisis basado en regiones | 672 |
| 9.7.1 | Regiones | 672 |
| 9.7.2 | Jerarquías de regiones para grafos de flujo reducibles | 673 |
| 9.7.3 | Generalidades de un análisis basado en regiones | 676 |
| 9.7.4 | Suposiciones necesarias sobre las funciones transformación | 678 |
| 9.7.5 | Un algoritmo para el análisis basado en regiones | 680 |
| 9.7.6 | Manejo de grafos de flujo no reducibles | 684 |
| 9.7.7 | Ejercicios para la sección 9.7. | 686 |
| 9.8 | Análisis simbólico | 686 |
| 9.8.1 | Expresiones afines de las variables de referencia | 687 |
| 9.8.2 | Formulación del problema de flujo de datos | 689 |
| 9.8.3 | Análisis simbólico basado en regiones | 694 |
| 9.8.4 | Ejercicios para la sección 9.8. | 699 |
| 9.9 | Resumen del capítulo 9 | 700 |
| 9.10 | Referencias para el capítulo 9 | 703 |
| 10 | Paralelismo a nivel de instrucción | 707 |
| 10.1 | Arquitecturas de procesadores | 708 |
| 10.1.1 | Canalizaciones de instrucciones y retrasos de bifurcación. | 708 |
| 10.1.2 | Ejecución canalizada | 709 |
| 10.1.3 | Emisión de varias instrucciones. | 710 |
| 10.2 | Restricciones de la programación del código. | 710 |
| 10.2.1 | Dependencia de datos | 711 |
| 10.2.2 | Búsqueda de dependencias entre accesos a memoria | 712 |
| 10.2.3 | Concesiones entre el uso de registros y el paralelismo | 713 |
| 10.2.4 | Ordenamiento de fases entre la asignación de registros y la programación de código | 716 |
| 10.2.5 | Dependencia del control | 716 |
| 10.2.6 | Soprote de ejecución especulativa | 717 |
| 10.2.7 | Un modelo de máquina básico. | 719 |
| 10.2.8 | Ejercicios para la sección 10.2. | 720 |
| 10.3 | Programación de bloques básicos | 721 |
| 10.3.1 | Grafos de dependencia de datos | 722 |
| 10.3.2 | Programación por lista de bloques básicos | 723 |
| 10.3.3 | Órdenes topológicos priorizados | 725 |
| 10.3.4 | Ejercicios para la sección 10.3. | 726 |
| 10.4 | Programación de código global | 727 |
| 10.4.1 | Movimiento de código primitivo | 728 |
| 10.4.2 | Movimiento de código hacia arriba | 730 |
| 10.4.3 | Movimiento de código hacia abajo. | 731 |
| 10.4.4 | Actualización de las dependencias de datos | 732 |
| 10.4.5 | Algoritmos de programación global | 732 |
| 10.4.6 | Técnicas avanzadas de movimiento de código | 736 |
| 10.4.7 | Interacción con los programadores dinámicos | 737 |
| 10.4.8 | Ejercicios para la sección 10.4. | 737 |

| | | |
|-----------|---|------------|
| 10.5 | Canalización por software. | 738 |
| 10.5.1 | Introducción | 738 |
| 10.5.2 | Canalización de los ciclos mediante software. | 740 |
| 10.5.3 | Asignación de recursos y generación de código | 743 |
| 10.5.4 | Ciclos de ejecución cruzada | 743 |
| 10.5.5 | Objetivos y restricciones de la canalización por software | 745 |
| 10.5.6 | Un algoritmo de canalización por software | 749 |
| 10.5.7 | Programación de grafos de dependencia de datos acíclicos | 749 |
| 10.5.8 | Programación de grafos de dependencia cíclicos | 751 |
| 10.5.9 | Mejoras a los algoritmos de canalización | 758 |
| 10.5.10 | Expansión modular de variables | 758 |
| 10.5.11 | Instrucciones condicionales | 761 |
| 10.5.12 | Soporte de hardware para la canalización por software | 762 |
| 10.5.13 | Ejercicios para la sección 10.5. | 763 |
| 10.6 | Resumen del capítulo 10 | 765 |
| 10.7 | Referencias para el capítulo 10 | 766 |
| 11 | Optimización para el paralelismo y la localidad | 769 |
| 11.1 | Conceptos básicos | 771 |
| 11.1.1 | Multiprocesadores | 772 |
| 11.1.2 | Paralelismo en las aplicaciones | 773 |
| 11.1.3 | Paralelismo a nivel de ciclo. | 775 |
| 11.1.4 | Localidad de los datos | 777 |
| 11.1.5 | Introducción a la teoría de la transformación afín. | 778 |
| 11.2 | Multiplicación de matrices: un ejemplo detallado | 782 |
| 11.2.1 | El algoritmo de multiplicación de matrices. | 782 |
| 11.2.2 | Optimizaciones | 785 |
| 11.2.3 | Interferencia de la caché. | 788 |
| 11.2.4 | Ejercicios para la sección 11.2. | 788 |
| 11.3 | Espacios de iteraciones. | 788 |
| 11.3.1 | Construcción de espacios de iteraciones a partir de anidamientos de ciclos | 788 |
| 11.3.2 | Orden de ejecución para los anidamientos de ciclos. | 791 |
| 11.3.3 | Formulación de matrices de desigualdades | 791 |
| 11.3.4 | Incorporación de constantes simbólicas | 793 |
| 11.3.5 | Control del orden de ejecución | 793 |
| 11.3.6 | Cambio de ejes | 798 |
| 11.3.7 | Ejercicios para la sección 11.3. | 799 |
| 11.4 | Índices de arreglos afines | 801 |
| 11.4.1 | Accesos afines | 802 |
| 11.4.2 | Accesos afines y no afines en la práctica | 803 |
| 11.4.3 | Ejercicios para la sección 11.4. | 804 |
| 11.5 | Reutilización de datos | 804 |
| 11.5.1 | Tipos de reutilización. | 805 |
| 11.5.2 | Reutilización propia. | 806 |

| | | |
|---------|--|-----|
| 11.5.3 | Reutilización espacial propia | 809 |
| 11.5.4 | Reutilización de grupo | 811 |
| 11.5.5 | Ejercicios para la sección 11.5 | 814 |
| 11.6 | Análisis de dependencias de datos de arreglos | 815 |
| 11.6.1 | Definición de la dependencia de datos de los accesos a arreglos | 816 |
| 11.6.2 | Programación lineal entera | 817 |
| 11.6.3 | La prueba del GCD | 818 |
| 11.6.4 | Heurística para resolver programas lineales enteros | 820 |
| 11.6.5 | Solución de programas lineales enteros generales | 823 |
| 11.6.6 | Resumen | 825 |
| 11.6.7 | Ejercicios para la sección 11.6 | 826 |
| 11.7 | Búsqueda del paralelismo sin sincronización | 828 |
| 11.7.1 | Un ejemplo introductorio | 828 |
| 11.7.2 | Particionamientos de espacio afín | 830 |
| 11.7.3 | Restricciones de partición de espacio | 831 |
| 11.7.4 | Resolución de restricciones de partición de espacio | 835 |
| 11.7.5 | Un algoritmo simple de generación de código | 838 |
| 11.7.6 | Eliminación de iteraciones vacías | 841 |
| 11.7.7 | Eliminación de las pruebas de los ciclos más internos | 844 |
| 11.7.8 | Transformaciones del código fuente | 846 |
| 11.7.9 | Ejercicios para la sección 11.7 | 851 |
| 11.8 | Sincronización entre ciclos paralelos | 853 |
| 11.8.1 | Un número constante de sincronizaciones | 853 |
| 11.8.2 | Grafos de dependencias del programa | 854 |
| 11.8.3 | Tiempo jerárquico | 857 |
| 11.8.4 | El algoritmo de paralelización | 859 |
| 11.8.5 | Ejercicios para la sección 11.8 | 860 |
| 11.9 | Canalizaciones | 861 |
| 11.9.1 | ¿Qué es la canalización? | 861 |
| 11.9.2 | Sobrerrelajación sucesiva (Successive Over-Relaxation, SOR): un ejemplo | 863 |
| 11.9.3 | Ciclos completamente permutables | 864 |
| 11.9.4 | Canalización de ciclos completamente permutables | 864 |
| 11.9.5 | Teoría general | 867 |
| 11.9.6 | Restricciones de partición de tiempo | 868 |
| 11.9.7 | Resolución de restricciones de partición de tiempo mediante el Lema de Farkas | 872 |
| 11.9.8 | Transformaciones de código | 875 |
| 11.9.9 | Paralelismo con sincronización mínima | 880 |
| 11.9.10 | Ejercicios para la sección 11.9 | 882 |
| 11.10 | Optimizaciones de localidad | 884 |
| 11.10.1 | Localidad temporal de los datos calculados | 885 |
| 11.10.2 | Contracción de arreglos | 885 |
| 11.10.3 | Intercalación de particiones | 887 |
| 11.10.4 | Reunión de todos los conceptos | 890 |
| 11.10.5 | Ejercicios para la sección 11.10 | 892 |

| | | |
|-----------|--|------------|
| 11.11 | Otros usos de las transformaciones afines | 893 |
| 11.11.1 | Máquinas con memoria distribuida | 894 |
| 11.11.2 | Procesadores que emiten múltiples instrucciones | 895 |
| 11.11.3 | Instrucciones con vectores y SIMD | 895 |
| 11.11.4 | Preobtención | 896 |
| 11.12 | Resumen del capítulo 11 | 897 |
| 11.13 | Referencias para el capítulo 11 | 899 |
| 12 | Análisis interprocedural | 903 |
| 12.1 | Conceptos básicos | 904 |
| 12.1.1 | Grafos de llamadas | 904 |
| 12.1.2 | Sensibilidad al contexto | 906 |
| 12.1.3 | Cadenas de llamadas | 908 |
| 12.1.4 | Análisis sensible al contexto basado en la clonación | 910 |
| 12.1.5 | Análisis sensible al contexto basado en el resumen | 911 |
| 12.1.6 | Ejercicios para la sección 12.1 | 914 |
| 12.2 | ¿Por qué análisis interprocedural? | 916 |
| 12.2.1 | Invocación de métodos virtuales | 916 |
| 12.2.2 | Análisis de alias de apuntadores | 917 |
| 12.2.3 | Paralelización | 917 |
| 12.2.4 | Detección de errores de software y vulnerabilidades | 917 |
| 12.2.5 | Inyección de código SQL | 918 |
| 12.2.6 | Desbordamiento de búfer | 920 |
| 12.3 | Una representación lógica del flujo de datos | 921 |
| 12.3.1 | Introducción a Datalog | 921 |
| 12.3.2 | Reglas de Datalog | 922 |
| 12.3.3 | Predicados intensionales y extensionales | 924 |
| 12.3.4 | Ejecución de programas en Datalog | 927 |
| 12.3.5 | Evaluación incremental de programas en Datalog | 928 |
| 12.3.6 | Reglas problemáticas en Datalog | 930 |
| 12.3.7 | Ejercicios para la sección 12.3 | 932 |
| 12.4 | Un algoritmo simple de análisis de apuntadores | 933 |
| 12.4.1 | Por qué es difícil el análisis de apuntadores | 934 |
| 12.4.2 | Un modelo para apuntadores y referencias | 935 |
| 12.4.3 | Insensibilidad al flujo | 936 |
| 12.4.4 | La formulación en Datalog | 937 |
| 12.4.5 | Uso de la información de los tipos | 938 |
| 12.4.6 | Ejercicios para la sección 12.4 | 939 |
| 12.5 | Análisis interprocedural insensible al contexto | 941 |
| 12.5.1 | Efectos de la invocación a un método | 941 |
| 12.5.2 | Descubrimiento del grafo de llamadas en Datalog | 943 |
| 12.5.3 | Carga dinámica y reflexión | 944 |
| 12.5.4 | Ejercicios para la sección 12.5 | 945 |
| 12.6 | Análisis de apuntadores sensible al contexto | 945 |
| 12.6.1 | Contextos y cadenas de llamadas | 946 |
| 12.6.2 | Agregar contexto a las reglas de Datalog | 949 |

| | | |
|----------|---|------------|
| 12.6.3 | Observaciones adicionales acerca de la sensibilidad | 949 |
| 12.6.4 | Ejercicios para la sección 12.6. | 950 |
| 12.7 | Implementación en Datalog mediante BDDs | 951 |
| 12.7.1 | Diagramas de decisiones binarios. | 951 |
| 12.7.2 | Transformaciones en BDDs. | 953 |
| 12.7.3 | Representación de las relaciones mediante BDDs | 954 |
| 12.7.4 | Operaciones relacionales como operaciones BDD | 954 |
| 12.7.5 | Uso de BDDs para el análisis tipo “apunta a” | 957 |
| 12.7.6 | Ejercicios para la sección 12.7. | 958 |
| 12.8 | Resumen del capítulo 12 | 958 |
| 12.9 | Referencias para el capítulo 12 | 961 |
| A | Un front-end completo | 965 |
| A.1 | El lenguaje de código fuente | 965 |
| A.2 | Main | 966 |
| A.3 | Analizador léxico. | 967 |
| A.4 | Tablas de símbolos y tipos | 970 |
| A.5 | Código intermedio para las expresiones | 971 |
| A.6 | Código de salto para las expresiones booleanas. | 974 |
| A.7 | Código intermedio para las instrucciones | 978 |
| A.8 | Analizador sintáctico | 981 |
| A.9 | Creación del front-end | 986 |
| B | Búsqueda de soluciones linealmente independientes | 989 |
| | Índice | 993 |