3.1 Plan de Estudios
Maestría en Ciencias en Computación

Departamento de Computación – CINVESTAV-IPN

SNP 2023

1. Análisis de pertinencia del programa

El programa de Maestría en Ciencias en Computación es un programa orientado a la investigación y ha sufrido cambios significativos a lo largo de su desarrollo. Siempre ha tenido una relación estrecha con las Líneas de Generación y Aplicación del Conocimiento (LGAC) que cultiva su planta académica. El plan de estudios actual ha tenido transformaciones importantes desde su creación. En el año 2000, se organizó por primera ocasión a través de cuatro LGAC: Fundamentos de la Computación e Inteligencia Artificial, Base de Datos y Sistemas de Información, Programación de Sistemas y Arquitectura de Computadoras. Las tres primeras LGAC tienen sus orígenes desde la creación del programa y han constituido la base para el desarrollo del programa orientado a la investigación.

En el año 2005, se reorganizó la LGAC de Arquitecturas de Computadoras para introducir los tópicos de Cómputo Reconfigurable y Criptografía que se habían venido desarrollando. También, se introdujo una LGAC relacionada con aspectos de Graficación, Visualización y Procesamiento de Imágenes. Posteriormente, el año 2006, se integró una nueva LGAC relacionada con Tecnologías de Información. En el año 2008 se hizo una actualización de la currícula del programa de maestría con el propósito de alcanzar el registro de profesiones ante la Secretaría de Educación Pública.

Finalmente, en el año 2015, con el objetivo de reflejar mejor la actividad de investigación de la planta académica del Departamento de Computación, se actualizaron las LGAC definidas como cuatro áreas principales de investigación, a saber: i) Teoría de la Computación, ii) Inteligencia Artificial, iii) Sistema de Cómputo y iv) Sistemas de Información. La actualización de las LGAC ha impactado de manera directa en los cursos impartidos, por lo tanto, se hacen revisiones constantes de sus contenidos, con el fin de mantenerlos actualizados.

El programa está dirigido a egresados de las carreras relacionadas con Computación como son: Ciencias de la Computación, Sistemas Computacionales, Ingeniería Computacional y en menor medida a los egresados de las licenciaturas en Informática. Por la orientación científica y tecnológica de este programa de maestría, egresados de las licenciaturas de Física, Matemáticas, Ingeniería Eléctrica, Electrónica y Mecatrónica también han resultado candidatos idóneos para cursar con éxito la Maestría en Ciencias en Computación. Los procesos de admisión a la maestría han tenido ajustes para incorporar estudiantes talentosos de los programas de licenciatura antes mencionados. Dichos ajustes se han realizado en la composición temática del examen de admisión el cual es presentado por todos los aspirantes. Reconociendo la diversidad de los perfiles de ingreso, el examen de admisión evalúa los fundamentos en ciencia e ingeniería que tienen los aspirantes, así como los conocimientos especializados que deben tener, de acuerdo con el perfil de egreso de la licenciatura.

La permanencia en el programa de maestría está regida por el Reglamento General de Estudios de Posgrado del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV-IPN). Un requisito que se aplica a todos los estudiantes del programa es que no deben

tener ninguna calificación reprobatoria (inferior a 7.0) y no deben tener un promedio inferior a 8.0. Se debe desarrollar una tesis y defenderla ante un comité de graduación. De manera particular, en el programa de maestría, se han hecho los ajustes para reducir los tiempos de graduación, a través del seguimiento de los trabajos de tesis desarrollados por los estudiantes durante el segundo año escolar. Se ha implementado un acuerdo interno que obliga, a los alumnos que no se gradúen a los 28 meses, a solicitar un permiso al Colegio de Profesores para presentar el examen de grado después de este tiempo. En caso de que el colegio determine no dar el permiso, el alumno queda dado de baja definitiva del programa. Por su parte, los profesores que no gradúen a sus alumnos en un lapso menor a 30 meses, tienen una sanción hacia las actividades académicas del posgrado.

Los seminarios de investigación cursados en el segundo año se han revisado y reforzado, a fin de que los estudiantes de maestría desarrollen tesis de calidad, de manera que sus resultados sean reportados en reuniones académicas e incluso en revistas con arbitraje. No obstante, la productividad científica reportada con los estudiantes de maestría puede ser mejorada, pues hasta la fecha no se ha establecido la publicación de un artículo, como requisito para obtener el grado de maestría.

A pesar de lo anterior, los resultados de los trabajos de tesis han sido satisfactorios, pues en varias ocasiones se ha obtenido lugares destacados en los concursos de tesis de maestría organizados por la Sociedad Mexicana de Inteligencia Artificial y por la Asociación Nacional de Instituciones de Educación en Informática (ANIEI). Asimismo, varios de los trabajos reportados con estudiantes de maestría en congresos internacionales han recibido reconocimientos de "*Best Paper Award*". También, se han obtenido títulos de patentes.

Los egresados del programa se desempeñan de manera adecuada en su etapa posterior a la obtención del grado. Las actividades desarrolladas por los egresados se dividen entre estudiantes de doctorado, docentes o académicos en instituciones de educación superior, prestadores de servicios de tecnologías de información y creadores de sus propias empresas.

2. Fundamentación, Objetivos y Metas

Las Ciencias de la Computación son parte importante de las tecnologías de la información y proporcionan el fundamento científico para el almacenamiento, aprovechamiento, transformación, procesamiento y transportación de tales tecnologías. A nivel nacional, el Programa Especial de Ciencia Tecnología e Innovación (PECiTI) ha reconocido a las tecnologías de información y comunicaciones como un área estratégica para el desarrollo de México y, debido a su gran cambio en periodos cortos, existe una demanda permanente por especialistas altamente calificados que afronten las necesidades del desarrollo nacional en dicho sector. Por otra parte, desde el año 2002 el gobierno federal, a través de la Secretaría de Economía, ha desarrollado el Programa para el Desarrollo de la Industria del Software (PROSOFT) para impulsar el sector de desarrollo de software en México, pues constituye una los sectores estratégicos para mejorar la competitividad de nuestro país.

Así también, diversos gobiernos estatales han desarrollado iniciativas regionales para el fomento del sector de desarrollo de software o de las tecnologías de información y comunicaciones. Una de las áreas de impulso, presentes en casi todos esos parques tecnológicos, es la referente a las Ciencias de la Computación en sus diferentes modalidades (e.g., tecnologías de la información, software, tecnologías de la información y las comunicaciones). Se han creado parques tecnológicos en Aguascalientes, Baja California, Chihuahua, Coahuila, CDMX, Durango, Edomex, Hidalgo, Jalisco, Michoacán, Morelos, Nuevo León, Sinaloa, Sonora, Tabasco y Tamaulipas.

En resumen, las tecnologías de la información se reconocen como un motor para el desarrollo de las economías basadas en conocimiento a niveles global, nacional y regional y permiten elevar la competitividad de las empresas en toda la cadena de producción. Se reconoce también que este sector demanda recursos humanos altamente especializados con habilidades y conocimientos requeridos a nivel especialización y posgrado y tiene una alta tasa de cambio científico y tecnológico. Finalmente, el sector mismo se reconoce como estratégico para el desarrollo nacional y tiene como característica la rapidez del cambio tecnológico. Nuestro programa de Maestría en Ciencias en Computación busca ofrecer soluciones a la demanda de posgraduados en las diversas especialidades de la Computación señaladas en el Programa Especial de Ciencia Tecnología e Innovación (PECiTI) 2020-2024.

Objetivo general

Preparar especialistas en las Ciencias de la Computación que conozcan y sepan aplicar la teoría, las metodologías y las técnicas más modernas de la disciplina. Asimismo, elevar el nivel académico de los egresados de nuestro programa de maestría, mediante la actualización de plan de estudios, el fortalecimiento de la infraestructura de laboratorios y el crecimiento de la planta de investigadores.

Objetivos específicos de los planes de estudio:

- Concebir un programa que conjugue, de forma armónica, los conocimientos teóricos con los prácticos y que esto se refleje en una formación integral de nuestros estudiantes.
- Ofrecer al estudiante un programa flexible que le permita enfocarse en el área específica de interés, sin abandonar otras áreas que le permitan una formación integral en Ciencias de la Computación.
- Implementar un programa adaptable y bien documentado para facilitar la revisión periódica de los contenidos de los cursos y la creación de nuevos cursos, de acuerdo a la demanda por parte de los estudiantes, de las empresas, así como a la demanda natural producida por la rápida evolución de las Ciencias de la Computación.
- Formar recursos humanos que tengan una visión global de las Ciencias de la Computación y que resuelvan problemas teórico-prácticos de diversa índole.

Metas:

- La formación de especialistas en las Ciencias de la Computación con una sólida base teórica y pericia práctica para la resolución de problemas.
- La formación de recursos humanos que puedan incrustarse en los diversos sectores de la vida nacional: investigación, enseñanza e industria.
- La formación de especialistas que aporten al desarrollo nacional con un fuerte sentido de ética, responsabilidad y compromiso con la sociedad.

3. Perfil de Ingreso

El programa de maestría está dirigido fundamentalmente, aunque no de forma exclusiva, a personas que han estudiado Ingeniería en Sistemas Computacionales, Ingeniería en Computación, Ingeniería en

Comunicaciones y Electrónica, Licenciatura en Informática, Licenciatura en Ciencias de la Computación, Licenciatura en Física y Matemáticas o áreas afines. Los conocimientos que se piden a los candidatos incluyen áreas como:

- Sistemas Operativos
- Estructura de Datos
- Compiladores
- Programación
- Arquitectura de Computadoras y Sistemas Digitales
- Lenguajes y Autómatas
- Redes de Computadoras
- Bases de Datos
- Ingeniería de Software
- Análisis Numérico y Teoría Elemental de Números y Probabilidad
- Diseño y Análisis de Algoritmos

Además, los aspirantes deben haberse titulado o tener una carta de pasante para cumplir con lo estipulado en el Artículo 18, Capítulo IX, del Reglamento General de Estudios de Posgrado (https://transparencia.cinvestav.mx/UNIDADENLACE2010/Reglamentogeneraldeestudiosdeposgrado2018.pdf) del CINVESTAV-IPN.

Las aptitudes que se consideran esenciales en el proceso de admisión son:

- Un alto sentido de compromiso y responsabilidad, debiendo dedicar el cien por ciento de su tiempo a la realización de sus actividades académicas.
- Inclinación hacia la investigación y el desarrollo tecnológico, con el fin de buscar soluciones con tecnología de punta para el bienestar social y desarrollo sustentable de su país.
- Una alta capacidad de análisis y proactividad para la solución de problemas y toma de decisiones que les permita proponer soluciones innovadoras a dichos problemas.
- Contar con la creatividad necesaria que les permita diseñar, innovar y proponer soluciones de mediano y largo alcance.
- Contar con una disposición para el trabajo en equipo.

Esas aptitudes son las que se evalúan durante todo el proceso de admisión, que consiste de un examen de conocimientos generales, un curso propedéutico y una entrevista.

4. Perfil de Egreso

Fomentamos el desarrollo de las capacidades analíticas de nuestros egresados y el equilibrio entre la resolución de problemas tecnológicos y la investigación básica. Así, nuestros egresados, dependiendo de la currícula que ellos seleccionen en la maestría, pueden optar por una formación completamente tecnológica, en donde adquieren los conocimientos para resolver problemas prácticos de desarrollo, o por una formación completamente teórica. En cualquiera de los dos casos nuestros egresados pueden continuar con un doctorado, dedicarse a la enseñanza superior o incorporarse al sector industrial (incluso, formar su propia empresa).

De esta forma, nuestro programa se enfoca en la formación de recursos humanos, encauzándose en tres

actividades sustantivas: a) la resolución de problemas tecnológicos y la práctica profesional, b) la investigación científica, y c) la docencia a nivel superior.

El enfoque de los estudios de maestría depende del estudiante, pudiendo ser de investigación o de aplicación en alguna de las áreas de la Computación que se mencionan más adelante.

5. Contenidos temáticos

El plan estratégico del programa de Maestría en Ciencias en Computación está basado en el Plan institucional del CINVESTAV-IPN en cuanto a los siguientes puntos:

- La solidez de las investigaciones científicas y tecnológicas de vanguardia logradas a través de la originalidad, del rigor de los estudios teóricos y experimentales y de la planta académica, manteniendo siempre el liderazgo de la institución.
- El desarrollo e impulso a la investigación científica básica, aplicada y tecnológica de excelencia a través de proyectos multidisciplinarios, interinstitucionales, de largo alcance y de alto impacto para la comunidad científica nacional e internacional y para la sociedad en general.
- La formación de recursos humanos de alto nivel académico, mediante la aplicación de programas de maestría y doctorado de la mejor calidad académica, en las disciplinas que actualmente se cultivan en la institución, promoviendo además el desarrollo de nuevos campos estratégicos.

Para ajustarse al Plan institucional del CINVESTAV-IPN, el programa de Maestría en Ciencias en Computación está orientado alrededor de cuatro ejes principales:

- **Investigación:** solución de problemas teóricos, propuesta y mejoramiento de algoritmos y procesos en áreas de vanguardia dentro de las Ciencias de la Computación.
- **Generación de tecnología de vanguardia:** a través de la solución de problemas específicos para diversos sectores de la sociedad.
- **Solución de problemas abiertos:** propuestas de solución a problemas existentes que se sustenten en el uso y conocimiento de tecnología de punta con una fuerte base teórica, brindando soluciones originales y de calidad.
- **Vinculación:** con el sector educativo, a través de conferencias por invitación de miembros de nuestro Colegio de Profesores y de nuestros estudiantes de posgrado y de la participación de comités de evaluación de programas y asesoramiento de colegios de profesores y estudiantes de otras instituciones. También, vinculación con empresas, a través del desarrollo de proyectos y de consultoría.

De esta forma, los ejes del programa de maestría garantizan: a) la formación de recursos humanos con una sólida base teórica en diversas áreas de vanguardia en las Ciencias de la Computación, b) el entrenamiento de especialistas capaces de dar seguimiento a soluciones propuestas, c) vinculación con el sector educativo, gracias a la participación de la planta académica en otros colegios dentro y fuera del CINVESTAV-IPN y a la formación de docentes que otras instituciones de educación superior en el país demandan y d) vinculación con las empresas para resolver problemas del mundo real que acarrean beneficios a la sociedad.

La excelencia de nuestros profesores y la exitosa inserción laboral de nuestros egresados en sectores académicos, gubernamentales y empresariales, tanto nacionales como internacionales, nos indican que

estamos cumpliendo con el plan institucional del CINVESTAV-IPN y con los ejes rectores del programa.

En el proceso de admisión, se analiza que los candidatos cuenten con una sólida formación y un verdadero interés por realizar estudios de maestría. En forma particular, los estudiantes del programa de maestría definen el perfil que quieren dar a sus estudios, al seleccionar sus cursos de acuerdo a sus intereses específicos. Sin embargo, dado que existe un conjunto de cursos formativos obligatorios, nuestros egresados poseen una sólida formación teórica y práctica en las diferentes áreas que conforman las Ciencias de la Computación.

6. Mapa curricular

El programa de maestría tiene una duración de dos años organizados en cuatrimestres e inicia en el cuatrimestre septiembre-diciembre de cada año. Así, el programa de estudios está dividido en dos fases cada una de un año escolar. Durante el primer año, el alumno debe tomar y aprobar un total de 12 cursos, cuatro por cuatrimestre. Cabe mencionar que el programa de maestría no maneja el sistema de créditos. Cada curso tiene una duración de 60 horas distribuidas a lo largo de un cuatrimestre y la escala de calificiones aprobatorias es entre 7.0 y 10. Durante el segundo año, el alumno debe desarrollar, con la asesoría de un profesor del Departamento de Computación, un proyecto de investigación (tesis), que el estudiante debe defender ante un jurado para obtener el grado de Maestro en Ciencias en Computación. Para ello, el alumno se inscribe en los cursos “Trabajo de Tesis” y “Seminario de Investigación”. Puede existir un co-director de tesis, cuya participación debe ser aprobada por el Colegio de Profesores, conformado por los miembros del Núcleo Académico (NA) y profesores visitantes.

Dada la influencia de la Computación en todas las áreas de conocimiento y con el fin de promover la multidisciplinariedad de la Computación, a lo más cuatro de los cursos pueden tomarse en otros Programas del CINVESTAV-IPN y a lo más dos cursos pueden tomarse en programas ofertados por otras instituciones. En otras palabras, la suma de los cursos acreditados por el estudiante en otros programas dentro y fuera del CINVESTAV-IPN no debe ser mayor a cuatro.

El enfoque de los estudios de maestría depende del estudiante, guiado por su asesor de estudios (durante el primer año) o director de tesis (durante el segundo año). El enfoque puede ser de investigación o de aplicación en alguna de las áreas de la Computación relacionadas con el programa.

Primer año: cursos

Los cursos a acreditar durante el primer año son seleccionados por cada estudiante y su asesor de estudios, i.e., un profesor del NA asignado al estudiante cuando ingresa al programa de maestría. La selección de los cursos busca que el estudiante adquiera una formación con los conocimientos esenciales de la Computación, con la mayor amplitud posible en las diferentes áreas de la Computación y con la mayor profundidad posible en las áreas relacionadas con el tema de tesis del estudiante y sus intereses de desarrollo profesional.

Los cursos están agrupados bajo un núcleo y cuatro áreas de especialidad. Todos los cursos del núcleo se ofrecen por lo menos una vez al año en los primeros dos cuatrimestres del año lectivo. Los cursos en las áreas de especialidad se ofrecen tomando en cuenta la demanda y la planta de profesores. Los cursos de cada área de especialidad se dividen en formativos y de especialización. Los formativos proporcionan amplitud de conocimientos y los de especialización dan profundidad en alguna LGAC.

Segundo año: desarrollo del trabajo de tesis

Durante el segundo año del programa, el estudiante seleccionará un tema de tesis propuesto por un profesor del Departamento de Computación, i.e., un miembro del NA. Puede existir un coasesor de tesis, que puede ser parte un profesor externo. El tema de investigación se somete a evaluación por un Comité de Aprobación conformado por dos miembros del NA. Asimismo, durante el segundo año, el alumno estará dedicado a los "Seminarios de Investigación" y a los desarrollos en laboratorio, los cuales corresponden a tres cursos de "Trabajo de Tesis" que se acreditan con la misma escala de calificación que los cursos del primer año. El alumno podrá hacer estancias industriales o académicas en otra institución de investigación.

Cada estudiante asesorado por su tutor académico deberá elegir ocho de los 12 cursos, de acuerdo a su área de especialización, aunque no necesariamente los ocho cursos deben ser de una misma LGAC. No todos los cursos se ofrecen en el mismo año escolar, sino que se abren dependiendo de la disponibilidad de los profesores y de la demanda de los estudiantes.

La trayectoria curricular del programa de maestria se presenta en la Tabla 1. En el primer año se toman 12 cursos, cuatro de núcleo y ocho optativas. En el segundo año, se evalúan tres cursos de "Seminario de Investigación" y tres cursos de "Trabajos de Tesis".

PRIMER AÑO

Primer Cuatrimestre Septiembre-Diciembre	Segundo Cuatrimestre Enero-Abril	Tercer Cuatrimestre Mayo-Agosto
Curso de Núcleo	Curso de Núcleo	Curso Optativo
Curso de Núcleo	Curso de Núcleo	Curso Optativo
Curso Optativo	Curso Optativo	Curso Optativo
Curso Optativo	Curso Optativo	Curso Optativo

SEGUNDO AÑO

Primer Cuatrimestre Septiembre-Diciembre	Segundo Cuatrimestre Enero-Abril	Tercer Cuatrimestre Mayo-Agosto
Seminario de Investigación I	Seminario de Investigación II	Seminario de Investigación III
Trabajo de Tesis I	Trabajo de Tesis II	Trabajo de Tesis III

Tabla 1. Trayectoria curricular del programa de Maestría en Ciencias en Computación

Cursos de Núcleo

Los cursos del núcleo comprenden los conocimientos básicos que cualquier egresado del programa de Maestría en Ciencias en Computación debe tener: 1) Diseño y Análisis de Algoritmos, 2) Programación

Avanzada, 3) Teoría de la Computación y 4) Arquitectura de Computadoras. Cada estudiante debe acreditar los cuatro cursos del núcleo, ya que el contenido de estos cursos es necesario por la diversidad del perfil de ingreso de nuestros estudiantes.

Cursos Formativos y de Especialización

Los cursos restantes para completar, al menos 12, se toman de las siguientes LGAC: Teoría de la Computación, Inteligencia Artificial, Sistemas de Cómputo y Sistemas de Información. Los cursos formativos y de especialización se presentan por LGAC en la Tabla 2.

En el Departamento de Computación, y antes cuando era la Sección de Computación del Departamento de Ingeniería Eléctrica, en concordancia con la libertad de cátedra e investigación se han ofrecido muchos más cursos de los aquí enlistados y algunos fueron impartidos tan solo una vez, así que si se revisara nuestro historial de cursos se encontraría ésos. Los aquí listados son vigentes.

Nivel	Teoría de la Computación	Inteligencia Artificial	Sistemas de Cómputo	Sistemas de Información
Formativos	Teoría de la Computación (N)	Inteligencia Artificial	Arquitectura de Computadoras (N)	Programación Avanzada (N)
	Diseño y Análisis de Algoritmos (N)	Procesamiento de Lenguaje Natural	Programación Orientada a Objetos	Base de Datos
	Matemáticas Discretas	Introducción a la Computación Evolutiva	Cómputo móvil	Lógica y Base de Datos
	Aritmética Computacional	Visión	Computación Paralela	Graficación
	Códigos y criptografia	Aprendizaje automático	Sistemas Distribuidos	Ingeniería de Software
	Geometría Computacional		Sistemas Operativos	Minería de Datos
	Lógica matemática		Programación Concurrente	Lenguajes de Programación
			Sistemas de tiempo real	
			Redes de Computadoras	

Nivel	Teoría de la Computación	Inteligencia Artificial	Sistemas de Cómputo	Sistemas de Información
Especialización	Optimización en Ingeniería	Visualización (TS)	Seguridad en Sistemas de Información	Pruebas de Confiabilidad de Software
	Optimización numérica	Sistemas de Agentes y Multiagentes (TS)	Diseño de Herramientas para Desarrollo de Sistemas Distribuidos	Sistemas Colaborativos Distribuidos
	Computibilidad y complejidad	Teoría de Juegos (TSIA)	Cómputo móvil (TS)	Sistemas distribuidos: eLearning (TS)
	Criptografía (TS)	Introducción a la Optimización Evolutiva Multiobjetivo (TSIA)	Introducción al cómputo reconfigurable	Interacción Hombre-Máquina (TS)
	Teoría de códigos (TS)	Aprendizaje automático (TS)	Sistemas operativos de tiempo real (TS)	Minería de Datos Avanzada (TS)
	Códigos Lineales (TS)	Redes Complejas y Aprendizaje Computacional (TS)	Cómputo Ubicuo	Análisis Estadísticos de Datos
	Computación Científica I (TS)	Sistemas de Soporte a la Toma de Decisiones (TS)	Programación de dispositivos moviles (TS)	Cómputación de alto rendimiento I (TS)
	Algoritmos en gráficas (TS)	Redes Neuronales Artificiales (TS)	Computación Sustentable (TS)	Cómputación de alto rendimiento II (TS)
		Aprendizaje profundo (TS)	Sistemas Ciber-físicos (TS)	Cómputación de alto rendimiento III (TS)
		Redes Complejas y Aprendizaje Computacional (TAIA)	Computación en Internet (TSSD)	Interfaces Conscientes del Contexto (TS)

Nivel	Teoría de la Computación	Inteligencia Artificial	Sistemas de Cómputo	Sistemas de Información
Especialización				Métodos de Evaluación de Interfaces de Usuario (TS)

(N) Curso de Núcleo, (TS) Tópicos Selectos, (TA) Tópicos Avanzados, (TSIA) Tópicos selectos en Inteligencia Artificial (TAIA) Tópicos Avanzados de Inteligencia Artificial, (TSSD) Tópicos Selectos de Sistemas Distribuidos

Tabla 2. Mapa Curricular de los cursos del programa de Maestría en Ciencias en Computación

7. Idioma

Ni al ingreso ni al egreso se solicita, como requisito, la acreditación del dominio de una lengua extranjera. No obstante, la bibliografía de nuestros cursos es generalmente en inglés y algunos cursos de especialización también se imparten en inglés, pero sólo los toman los alumnos que tienen dominio del idioma.

8. Actualizaciones al plan de estudios (última actualización diciembre de 2022)

Las Ciencias de la Computación son una de las áreas de investigación que evolucionan más rápido, por lo que se necesita un programa de maestría que siga esa evolución. Nuestro programa, además de ser flexible para el estudiante, permite la creación y proposición de nuevos cursos, conforme éstos sean demandados por los alumnos o por los avances que se van dando en la evolución de la disciplina. De esta forma, podemos decir que nuestro programa de estudios se encuentra en permanente actualización. De no hacerlo así, nuestro programa se volvería obsoleto muy rápidamente. Los siguientes cursos fueron creados en los últimos dos años académicos de operación del programa:

- Tópicos Selectos de Métodos de Evaluación de Interfaces de Usuario
- Tópicos Selectos de Interfaces Conscientes de Contexto
- Tópicos Selectos de Aprendizaje Automático
- Tópicos Selectos de Algoritmos en Gráficas
- Tópicos Selectos de Computación Sustentable
- Tópicos Avanzados de Inteligencia Artificial: Redes Complejas y Aprendizaje Computacional
- Tópicos Selectos de Sistemas Ciber-físicos
- Tópicos Selectos de Sistemas Distribuidos: Computación en Internet

9. Opciones de graduación

La única opción de graduación con la que contamos es el examen de grado. Durante el segundo año de maestría, el estudiante selecciona un tema de tesis de entre las propuestas de los miembros del NA; es también posible que el estudiante proponga un tema de tesis a algún profesor. El trabajo de tesis se desarrolla durante un año, al término del cual el estudiante deberá presentar su trabajo en forma escrita y oral en una sesión abierta al público y con un jurado compuesto por, al menos, tres miembros del NA

(es posible tener un profesor externo, siempre y cuando los profesores del NA conformen la mayoría del jurado).

10. Líneas de generación y/o aplicación del conocimiento

LGAC 1: Teoría de la Computación (CA-TC): en esta línea de investigación se trata de conocer las capacidades y limitaciones fundamentales de las computadoras. Para ellos se incluyen temas como Teoría de complejidad, cuyo objetivo principal es clasificar problemas de acuerdo a un grado de dificultad); teoría de computabilidad, donde el objetivo principal clasificar problemas que son solubles o no solubles, y finalmente los modelos computacionales. La orientación principal en esta área incluye teoría de la complejidad, teoría de computabilidad, lógica matemática, criptografía, seguridad, teoría de juegos, diseño y análisis de programas y lenguajes de programación, análisis numérico, optimización y geometría computacional.

LGAC 2: Inteligencia Artificial (CA-IA): la inteligencia artificial es el estudio de soluciones para problemas que son difíciles o impacticos de resolver con métodos tradicionales. Los modelos de razonamiento usados para el desarrollo de sistemas inteligentes. El área de Computación Evolutiva, considerada como parte de esta línea, se refiere al uso de sistemas bioinspirados para la solución de problemas computacionales difíciles. Esta área ha tenido un desarrollo reciente importante y se espera que sea una de las que logre mayor impacto dentro del grupo de computación del CINVESTAV. Las áreas específicas son teoría de juegos, redes neuronales, procesamiento de lenguaje natural, representación de conocimiento y razonamiento, métodos de búsqueda y *machine learning*.

LGAC 3: Sistemas de Cómputo (CA-SC): en esta área se estudia el diseño y desarrollo de software para administrar los recursos de sistemas de cómputo y para desarrollar software de aplicación. Es de destacar en esta área la importancia cada vez mayor de los mecanismos de seguridad informática a nivel de computadoras y redes de computadoras, los cuales requieren tomar como base estrategias generales para integrar soluciones ad hoc para un problema específico. Áreas específicas: Redes de computadoras, sistemas para arquitecturas paralelas y distribuidas, control de procesos y sistemas de tiempo real.

LGAC 4: Sistemas de Información (CA-SI): la organización, almacenamiento, comunicación y presentación de la información es fundamental. Esta línea comprende el desarrollo e integración de sistemas de software basado en la descomposición funcional, el desarrollo de herramientas de software y técnicas de recuperación de información. Las áreas específicas son Bases de datos, recuperación de información, interfaz hombre-máquina, minería de datos, redes de Petri, *eLearning*, ingeniería de software y graficación.

11. Programa integrado con el Doctorado en Ciencias en Computación

El programa de Maestría en Ciencias en Computación está integrado con el Doctorado en Ciencias en Computación y ambos se imparten en el edificio del Departamento de Computación de CINVESTAV-IPN Unidad Zacatenco. Ambos programas tienen las mismas cuatro LGAC. Por lo general, los alumnos del programa de maestría llevan los cursos de núcleo y formativos, mientras que los alumnos del programa de doctorado llevan los cursos de especialización. Nuestros programas incluyen varios cursos de tópicos selectos en diferentes subdisciplinas de la Computación, con el fin de ofrecer a los alumnos de doctorado un mayor nivel de profundidad en temas específicos. Con el fin de facilitar a los estudiantes la transición del programa de maestría al programa de doctorado, nuestros egresados de maestría quedan

exentos del examen escrito de conocimientos generales, que es un requisito de ingreso al programa de doctorado, siempre y cuando no hayan transcurrido más de tres años de haber realizado el examen de obtención de grado de maestría.

12. Descripción de los cursos

A continuación, se presentan los contenidos detallados por cada curso ofertado en el programa.

Arquitectura de Computadoras

Objetivo

Revisar la organización y arquitectura de los sistemas de cómputo modernos que permiten mejorar su rendimiento.

Descripción

Los avances en el rendimiento de los procesadores modernos son dramáticos. Aun cuando buena parte del rendimiento actual se debe a los avances en la tecnología de computadoras, esto mismo ha permitido que la arquitectura de los procesadores evolucione y se pueden ejecutar más y mejores funciones directamente sobre un procesador. La disponibilidad a bajo costo de los microprocesadores, hace que el estudio de la arquitectura de computadoras sea necesario para aquel interesado en conocer cómo explotar al máximo el rendimiento de los procesadores actuales. En el curso se revisa la organización de las computadoras modernas y sus diferentes componentes. Se revisa la arquitectura de los procesadores modernos, su conjunto de instrucciones y la jerarquía de memoria sobre la cual estos han sido diseñados.

Se revisan los aspectos más relevantes de la arquitectura de computadoras los cuales le permiten ofrecer mejores rendimientos. Después de revisar los aspectos para evaluar el rendimiento de un procesador, se revisan los avances en el diseño de conjuntos de instrucciones. Posteriormente, se revisa la organización de la jerarquía de memoria y los diferentes aspectos sobre la ejecución paralela de varias instrucciones.

Contenido

a. Conceptos básicos.
    a. Diseño lógico
    b. Aritmética computacional
    c. Tipos de dispositivos computacionales

b. Fundamentos de arquitecturas de computadoras.
    a. Clases de computadoras
    b. Taxonomía de Flynn
    c. Definición de una Arquitectura de Computadora
    d. Tendencias tecnológicas

c. Diseño de instrucciones
    a. Tipos de instrucciones
    b. Modos de direccionamiento de memoria
    c. Control de flujo
    d. Pipeline
    e. Predicciones de salto
    f. Paralelismo a nivel de instrucciones

d. Diseño de jerarquía de memoria.

    a. Optimizaciones del rendimiento de memoria
    b. Tecnología de memoria y optimización
    c. Protección: Memoria virtual y máquinas virtuales
    d. Casos de estudio.
5. Arquitectura multicore
    a. Paralelismo a nivel hilo
    b. Arquitecturas de memoria centralizada compartida
    c. Memoria compartida distribuida y coherencia
    d. Arquitectura de cache

6. Procesamiento vectorial y arquitectura de GPU
    a. Arquitectura vectorial
    b. Conjunto de instrucciones SIMD para multimedia
    c. Unidades de procesamiento gráfico
    d. Arquitectura many-core heterogéneas (asimétricas)
    e. Detección y mejora de paralelismo a nivel ciclo

Bibliografía básica

1. Hennessy, John L., and David A. Patterson, Computer Architecture: A Quantitative Approach: 6a Edition. Morgan Kauffmann Publishers. (7 Diciembre 2017).
2. Harris, David, and Sarah Harris. Digital design and computer architecture. Elsevier, 2012.
3. Solihin, Yan. Fundamentals of parallel computer architecture. Solihin Publishing Consulting LLC, 2009.
4. Jean-Loup Baer. Microprocessor Architecture, From Simple Pipelines to Chip Multiprocessors. Cambridge University Press, 2009.

Bibliografía complementaria

5. Shen, John Paul, and Mikko H. Lipasti. Modern processor design: fundamentals of superscalar processors. Waveland Press, 2013.
6. Iannucci, Robert A., et al., eds. Multithreaded computer architecture: A summary of the state of the art. Vol. 281. Springer Science Business Media, 2012.
7. Patterson, David A., and John L. Hennessy. Computer organization and design RISC-V edition: the hardware/software interface. Morgan Kaufmann; 1er edición (27 de abril 2017).

Diseño y análisis de algoritmos

Objetivo

Presentar las técnicas para analizar y diseñar algoritmos y revisar la teoría computacional relacionada con la clasificación de problemas. En este curso se revisará el proceso de análisis de algoritmos, así como las técnicas utilizadas para diseñar algoritmos eficientes. En la primera parte se introducirán algunos conceptos matemáticos necesarios para el análisis de algoritmos. Se revisarán los modelos computacionales más utilizados y se definirá de manera breve lo que se entiende por complejidad computacional. En la segunda parte se ejemplificará la complejidad de los algoritmos mediante el análisis de algoritmos típicos como ordenamiento, búsquedas, algoritmos sobre gráficas, etc. En la tercera parte se presentarán técnicas de diseño de algoritmos generales como programación dinámica, algoritmos ávidos y métodos *branch-and-bound*. Finalmente, en la cuarta parte se presentarán algunos resultados que determinan las clases de complejidad. Se introducirá la clasificación de los problemas de decisión, los problemas difíciles y los problemas completos, los problemas polinomiales y no-polinomiales.

Contenido

1. Introducción
   1. Motivación
   2. Algoritmos

Parte uno: Fundamentos

2. Nociones matemáticas y terminología
   1. Análisis de algoritmos
   2. Diseño de algoritmos
   3. Ejemplos básicos

3. Crecimiento de funciones
   1. Notación asintótica
   2. Las funciones más comunes

4. Divide y vencerás y métodos para resolver recurrencias
   1. Dos problemas que se resuelven usando divide y vencerás:
      1. El problema del sub-arreglo máximo
      2. Multiplicación de matrices: algoritmo de Strassen
   2. El método de sustitución para resolver recurrencias
   3. Árbol de recursión para resolver recurrencias
   4. El método maestro

5. Análisis probabilístico y algoritmos aleatorizados
   1. Un problema que se resuelve usando algoritmos aleatorizados
   2. Variables indicadoras
   3. Algoritmos aleatorizados y su análisis

Parte dos: Ordenación y Estadísticas de órden

6. Algoritmos básicos de ordenación
   1. Heapsort
   2. Quicksort

7. Ordenación en tiempo lineal
   1. Cota inferior de ordenación
   2. Counting sort, bucket sort, radix sort

8. Medianas y otras estadísticas de orden
    1. Mínimo y máximo
    2. i-ésima estadística de órden en tiempo línea

Parte tres: Estructura de datos

9. Árboles binarios de búsqueda (Binary Search Trees)
    1. Operaciones básicas
    2. BST construidos aleatoriamente

10. Red-black Trees
    1. Propiedades
    2. Operaciones básicas

11. Estructura de datos dinámicas (augmenting data structures)
    1. Estadísticas de orden dinámicas
    2. Cómo aumentar una estructura de datos
    3. Una estructura de datos dinámica: Árboles de intervalos

Parte cuatro: Técnicas avanzadas de diseño y análisis

12. Programación dinámica
    1. Dos problemas que se resuelven usando programación dinámica:
        1. Rod cutting
        2. Multiplicación de cadenas de matrices
    2. El método de la programación dinámica
    3. Subsecuencia común más larga

13. Algoritmos glotones (greedy algorithms)
    1. Un problema que se resuelve usando algoritmos glotones: el problema de la selección de actividades
    2. Los algoritmos glotones
    3. Códigos de Huffman.

14. Análisis amortizado (*opcional)
    1. Análisis agregado (aggregate analysis)
    2. El método de la contabilidad (the accounting method) y el método del potencial (the potential method)
    3. Tablas dinámicas

Parte cinco: Algoritmos en gráficas

15. Algoritmos elementales en gráficas
    1. BFS, DFS
    2. Topological sort
    3. Componentes conexas

16. Árboles generadores de peso mínimo
    1. Propiedades
    2. Algoritmo de Prim y Kruskal

17. Caminos más cortos
    1. Caminos más cortos desde una sola fuente
        1. Algortimo de Bellman-Ford y Dijkstra
    2. Caminos más cortos para todas las parejas
        1. Algoritmo de Floyd-Warshall y de Johnson

Bibliografía básica

1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. *Introduction to Algorithms, Third Edition* (3rd ed). The MIT Press.
2. Jon Kleinberg and Eva Tardos. 2005. *Algorithm Design.* Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.
3. Alfred V. Aho, and John E. Hopcroft, 1974 *The Design and Analysis of Computer Algorithms*. (1st ed.) Addisson-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Bibliografía complementaria

4. Robert Sedgewick, Kevin Wayne. 2011. *Algorithms, Fourth Edition* (4th ed.). Addison-Wesley Professional.
5. George Heineman, Gary Pollice, and Stanley Selkow. 2008. *Algorithms in a Nutshell*. O'Reilly Media, Inc.
6. Thomas H. Cormen. 2013. *Algorithms Unlocked* (1 ed). The MIT Press.
7. Peter Brass. 2008. *Advanced Data Structures* (1 ed.). Cambridge University Press, New York, NY, USA.

Programación Avanzada

Objetivo

En este curso se revisan distintos paradigmas de programación: estructurado, funcional, y orientado a objetos entre otros. Se discuten estrategias para optimizar la implementación de algoritmos, dependiendo del paradigma que se trabaje. Una parte de curso hace revisión de manejo de apuntadores (a datos, estructuras y a funciones), y los equivalentes en otros paradigmas de programación (selectores y tablas de funciones). En la parte de Programación Orientada a Objetos se discuten algunos de los patrones de diseño más utilizados, como pueden ser *oberver, MVC,* o *decorator*, por mencionar algunos.

Contenido

I. Conceptos básicos.
- I.1 Proceso de desarrollo de un programa.
- I.2 Tipos de datos abstractos.
- I.3 Organización de código.
- I.4 Funciones.
- I.5 Punteros y manejo de memoria.
- I.6 Fugas de memoria.

II. Estructuras de datos.
- II.1 Pilas, colas, listas, listas doblemente ligadas.
- II.2 árboles: árboles, binarios, recorridos, balanceo, búsqueda.
- II.3 Algoritmos de búsqueda, ordenamiento y hash.

III STL (Standar Template Library)
- III.1 Vectores
- III.2 Iteradores.
- III.3 Vectores bidimensionales.
- III.4 Colas y listas.

IV Scripts
- IV.1 Shell.
- IV.2 Perl.
- IV.3 Python.
- IV.4 Ruby.

Bibliografía

a. Reema Thareja. Data Structures Using C. Oxford University Press, Inc., New York, NY, USA. 2011.
b. Kernighan, Brian W., Rob Pike. El entorno de programaciÛn UNIX Prentice-Hall Hispanoamericana, 1987.
c. Kernighan, Brian W., and Dennis M. Ritchie. El lenguaje de programaciÛn C. Pearson Educación, 1991.
d. Wall, Larry, Tom Christiansen, and Jon Orwant. Programming perl. O'Reilly Media, Inc., 2004.
e. Sanner, Michel F. Python: a programming language for software integration and development. J Mol Graph Model 17.1 (1999): 57-61.

f. Flanagan, David, and Yukihiro Matsumoto. The ruby programming language. O'Reilly Media, Inc.", 2008.
g. Marquez, Francisco. Unix-Programacion Avanzada. Alfaomega Grupo Editor, 2005.

Lógica Matemática

Objetivos

Presentar los conocimientos básicos de Lógica Matemática para poder trabajar en Programación Lógica, en Inteligencia Artificial y, en general, en Ciencias de la Computación.

Descripción

En este curso se pretende revisar las ideas principales de la Lógica Matemática con miras a su aplicación. Esto incluye un tratamiento del cálculo de proposiciones, de la lógica de primer orden, de una introducción a ciertas lógicas modales y polivalentes y al uso de PROLOG.

El curso es asímismo una introducción a PROLOG. Las nociones del lenguaje serán introducidas a lo largo del curso. PROLOG es pues el lenguaje para la realización de prácticas.

Contenido

1.- Preliminaries
   1. Logic Formulas
   2. Semantics of Formulas
   3. Models and Logical Consequence,
   4. Logical Inference
   5. Substitutions

2.- Definite
   a. Logic Programs
   b. Definite Clauses,
   c. Definite Programs and Goals
   d. The Least Herbrand Model
   e. Construction of Least Herbrand Models

3.-SLD-Resolution:
   1. Informal Introduction
   2. Unification
   3. SLD-Resolution
   4. Soundness of SLD-resolution
   5. Completeness of SLD-resolution
   6. Proof Trees

4.-Negation in Logic Programming:
   a. Negative Knowledge
   b. The Completed Program
   c. SLDNF-resolution for Definite Programs

d. General Logic Programs
e. SLDNF-resolution for General Programs
f. Three-valued Completion
g. Well-founded Semantics

5.-Towards Prolog:

1. Cut and Arithmetic: Cut: Pruning the SLD-tree
2. Built-in Arithmetic

6.-Logic and Databases:

a. Relational Databases,
b. Deductive Databases,
c. Relational Algebra vs. Logic Programs,
d. Logic as a Query-language
e. Special Relations,
f. Databases with Compound Terms
g. Programming with Recursive Data Structures: Recursive Data Structures,
h. Lists,
i. Difference Lists

7.-Amalgamating Object- and Meta-language:

a. What is a Meta-language?,
b. Ground Representation,
c. Nonground Representation,
d. The Built-in Predicate
e. clause/2, The Built-in Predicates assert {a,z}/1,
f. The Built-in Predicate retract/1

8.-Logic and Expert Systems:

a) Expert Systems,
b) Collecting Proofs,
c) Query-the-user,
d) Fixing the Car (Extended Example)

9.-Logic and Grammars:

a) Context-free Grammars,
b) Logic Grammars,
c) Context-dependent Languages,
d) Definite Clause Grammars (DCGs),
e) Compilation of DCGs into Prolog

10.-Searching in a State-space:

1. State-spaces and State-transitions,
2. Loop Detection,
3. Water-jug Problem (Extended Example),
4. Blocks World (Extended Example),
5. Alternative Search Strategies

11.-Logic Programming and Concurrency:

1) Algorithm = Logic + Control,
2) And-parallelism,
3) Producers and Consumers,
4) Don't Care Nondeterminism,
5) Concurrent Logic Programming

12.-Logic Programs with Equality:

a. Equations and E-unification,
b. More on E-unification,
c. Logic Programs with Equality

13.-Constraint Logic Programming:

a. Logic Programming with Constraints,
b. Declarative Semantics of CLP,
c. Operational Semantics of CLP,
d. Examples of CLP-languages

14.-Query-answering in Deductive Databases:

e. Naive Evaluation,
f. Semi-naive
g. Evaluation,
h. Magic Transformation,
i. Optimizations

Bibliography

1.- Ulf Nilsson and Jan Maluszynski, Logic, Programming and Prolog (2ed), John Wiley & Sons Ltd., 2000, http://www.ida.liu.se/~ulfni/lpp/

2.- Guillermo Morales-Luna: Lógica Matemática (Un enfoque computacional), Edición electrónica del autor, 2002. En PDF y en HTML: http://delta.cs.cinvestav.mx/~gmorales/LogicaComputacional/alog01/alog01.html

Teoría de la Computación

Objetivos:
Presentar como mero recordatorio, y acaso profundizar un poco en sus aspectos algebraicos, a la Teoría de Autómatas y sus relaciones con las Gramáticas Formales.

Descripción:
En este curso se revisa las ideas principales de los autómatas, finitos en sus estados pero potencialmente infinitos en sus memorias, y sus correspondencias con los lenguajes formales.
El curso comprende muchas prácticas de programación. Cada estudiante está en libertad de escoger el ambiente de programación que considere más adecuado para la realización de sus tareas. En clase se ilustrará algunos procedimientos mediante el calculador simbólico xMaxima.

Contenido

1. Introducción
    1. Autómatas, Computación y Complejidad
    2. Nociones Matemáticas y Terminología
    3. Conjuntos
    4. Funciones, relaciones
    5. Pruebas, tipos de pruebas

Parte uno. Autómatas y Lenguajes
2. Lenguajes Regulares
    a. Autómatas finitos
    b. Autómatas no-deterministas
    c. Expresiones regulares y lenguajes
    d. Lenguajes no-regulares

3. Lenguajes Libres de Contexto
    a) Gramáticas libres de contexto
    b) Autómatas de pila (pushdown automaton)
    c) Lenguajes no-libres de contexto

Parte dos. Teoría de la Computabilidad
4. La Tesis de Church-Turing
1. Máquinas de Turing
2. Variantes de las Máquinas de Turing
3. Definición de algoritmo. El problema de Hilbert

5. Decibilidad
    a) Lenguajes decidibles
    b) Problema de la detención (Halting problema)
    c) Problemas indecidibles

Parte tres. Teoría de la Complejidad
6. Complejidad como función del tiempo
1. Medidas de complejidad
2. La clase de P y la clase de NP

3. NP-completitud
4. Algunos problemas NP-completos

7. Complejidad como función del espacio
1. Teorema de Savitch
2. La clase PSPACE y PSPACE-completitud
3. Las clases N, NL y NL-completitud
4. Jerarquía de la complejidad

Bibliografía

Libros recientes de texto

1.- Keith Cooper, Linda Torczon, Engineering a Compiler, 2nd Edition, Morgan Kaufmann, 2011

2.-Jim Hefferon, Theory of Computation, 2019

3.-Peter Linz, An Introduction to Formal Languages and Automata 6th Edition, Jones & Bartlett Learning; 6th edition, 2016

4.-A.M. Padma Reddy, Finite Automata And Formal Languages : A Simple Approach, Cengage, 2019

5.-Jean-Éric Pin, Mathematical Foundations of Automata Theory, Lecture notes LIAFA, Université Paris, 2010

6.-Dana Richards, Henry Hamburger, Logic And Language Models For Computer Science, 3rd Edition, WSPC, 2017

7.-Michael Sipser, Introduction to the Theory of Computation, 3rd edition, Cengage Learning, 2012. El autor ofrece su curso aquí, con temario y láminas.

8.- Guillermo Morales: Principios de autómatas finitos, México, 2000

Bibliografia de texto clásicos

1.-Aho, Ullman: Foundations of computer science, W. H. Freeman & Co., 1992
2.-Arbib: The algebraic theory of machines, languages and semigrups, Academic Press, 1968
3.-Arbib, Kfoury, Moll: A basis for theoretical computer science, Springer-Verlag, 1981
4.-Conway: Regular algebra and finite machines, Chapman & Hall, 1971
5.- Denning, Dennis, Qualitz: Machines, languages and computation, Prentice-Hall, 1978
6.-Eilenberg: Automata, languages, and machines, Volume A, Academic Press, 1974
7.-Ginzburg: Algebraic theory of automata, Academic Press, 1968
8.-Harrison: Introduction to switching and automata theory, McGraw-Hill Book company, 1965
9.-Harrison: Introduction to formal languages theory, Addison Wesley, 1978 Hopcroft, Ullman: Introduction to automata theory, languages and computation, Addison-Wesley, 1979
10.-Kfoury, Moll, Arbib: A programming approach to computability, Springer-Verlag, 1982
11.- Kohavi: Switching and finite automata theory, McGraw-Hill, 1979
12.-Kuich, Salomaa: Semirings, automata, languages, Springer-Verlag, 1985
13.-Lewis, Papadimitiou: Elements of the theory of computation, Prentice Hall, 1981
14.-Rytter: 100 execises in the theory of automata and formal languages, Research report 99, Dept. Comp. Sci., University of Warwick, 1987
15.-Rogers: Theory of recursive functions and effective computability, McGraw-Hill, 1967
16.-Salomaa: Automata theory, Pergamon Press, 1969 Salomaa: Formal languages, Academic Press, 1973

Bases de datos

Objetivo

Presentar diversos modelos de datos que son abstracciones matemáticas para representar la información del mundo real en datos y conocimiento. El curso cubre también los aspectos de la organización física de los datos, con detalles de implantación para cada uno de los modelos lógicos.

Descripción

Los diversos modelos son unificados mediante el modelo ente-vinculo de Chen que incorpora importante información semántica correspondiente al mundo real. Tomando como punto de partida el modelo de Chen, se tratan los modelos semánticos de datos que incluyen técnicas de Representación de Conocimiento. Finalmente, considerando la corriente de extender los modelos basados en entidades y abstracción en base de datos, tratamos el enfoque Orientado a Objetos.

Contenido

1. Introducción a las bases de datos
    a. Retrospectiva y perspectiva del desarrollo de las bases de datos
    b. Representación y semántica
2. El modelo relacional de datos
    a. Esquemas de relación y formas normales
    b. Lenguajes del modelo relacional
    c. Completitud relacional.
3. Diseño de bases de datos
    a. El proceso de diseño de base de datos
    b. El modelo entidad-vinculo-extendido
    c. Teoría de diseño
4. Lenguajes y operaciones
    a. Lenguajes EVEX
    b. Lenguaje visual de dominios
    c. Lenguaje LIDA
    d. Aplicaciones PosgresSQL
5. Base de datos geográficas
    a. El modelo objeto-relacional/entidad-vinculo
    b. Consultas espaciales y temáticas
    c. Análisis de datos espaciales
    d. Aplicación: Grass y Postgis
6. Base de datos activas
    a. Análisis y modelos
    b. Arquitectura de sistemas de base de datos activas

Bibliografía básica

1. C.J. Date. Introducción a los Sistemas de Bases de Datos, Vol. I, 6a. Edicion, Addison Wesley.
2. H. S. Korth Y A. Silberschatz, Fundamentos de Bases de Datos. Mc Graw Hill.

3. Garcia-Molina, Hector; Ullman, Jeffrey D.; Widom, Jennifer D. (Prentice Hall) Database Systems. Stanford. ISBN: 0130980439.
4. Jeffrey D. Ullman - Jennifer Widom. Introduccion a las Bases de Datos Prentice Hall. ISBN: 9701702565.

Bibliografía

1. C.J. Date. Database In Depth. O'reilly & Associates. ISBN: 0596100124. (May 2005).
2. Rex Porbasas Flejoles, Database Theory and Application, Arcler Press, 2017
3. Alex Kriegel And Boris M. Trukhnov. Sql Bible. Editorial: Wiley; Edition (April 1, 2003) ISBN: 0764525840
4. Toby J. Teorey, Sam S. Lightstone, And Tom Nadeau. Database Modeling and Design: Logical Design. Morgan Kaufmann; 4 Edition (September 6, 2005). ISBN: 0126853525
5. Houlette. Fundamentos de SQL. 1A Edición. Editorial Mcgraw-Hill. 2003 ISBN: 9701038959.

Códigos y Criptografía

Objetivo

El alumno aprenderá algoritmos de criptografía de clave secreta. Se abordarán las bases matemáticas, la seguridad demostrable, las implementaciones eficientes y seguras tanto en software como en hardware, así como los ataques y protecciones

Requisitos: Deseable conocimiento de álgebra moderna, probabilidad y variables aleatorias.

Contenido

1. Antecedentes matemáticos
    1. Conjuntos
    2. Grupos
    3. Semigrupos
    4. Anillos
    5. Campos
2. Criptosistemas clásicos
    a. Cifrado por sustitución
    b. Cifrado de César
    c. Cifrado Afín
    d. Ataques
3. Funciones y permutaciones
    a. Funciones y permutaciones aleatorias
    b. Funciones y permutaciones pseudo-aleatorias
4. Cifradores por bloque
    1.1. Seguridad
    1.2. Redes de Feistel
        1.2.1. DES
        1.2.2. Simon y Speck
    1.3. Redes de sustituciones y permutaciones
        1.3.1. AES
        1.3.2. Present y Gift
    1.4. Modos de operación clásicos (privacidad)
5. Cifradores por flujo de datos
    1.1 Seguridad
    1.2 Generadores de números aleatorios
    1.3 Registros de corrimiento con retroalimentación
        i. Trivium y Grain
    1.4 RC4
    1.5 Salsa20
6. Funciones hash
    a. Seguridad
    b. Construcción de Merkle-Damgard
    c. Estándares de hash: sha1 y sha2
    d. Permutaciones públicas (Funciones esponja)
        i. Keccak (sha3)
7. Códigos de autenticación de mensajes
    a. Seguridad
    b. Basados en funciones hash
        i. HMAC

    c. Basados en cifradores por bloque
        i. Funciones hash universales
        ii. GMAC
        iii. CBCMAC
        iv. PMAC
    d. Basados en funciones esponja
8. Cifrado autenticado
    a. Seguridad
    b. Basados en cifradores por bloque
        a. OCB
        b. GCM
        c. LOCUS y LOTUS
        d. SIV y ESTATE
        e. ElmD, COPA y COLM
    c. Basados en cifradores por flujo de datos
        a. Trivia-ck
        b. Acorn
    d. Basados en funciones esponja
        a. ASCON
        b. Oribatida

Bibliografía

1. Elena Andreeva, Andrey Bogdanov, Nilanjan Datta, Atul Luykx, Bart Mennink, Mridul Nandi, Elmar Tischhauser and Kan Yasuda. COLM v1. Final portfolio CAESAR Competition. September 15, 2016. Available online: https://competitions.cr.yp.to/round3/colmv1.pdf (Accessed 9th February 2021).
2. Mihir Bellare and Phillipe Rogaway. Introduction to Modern Cryptography. Available online:https://www.cs.ucdavis.edu/~rogaway/classes/227/fall03/book/toc.pdf (accessed February 3, 2021)
3. Guido Bertoni, Joan Daemen, Michaël Peeters and Gilles Van Asshe. The Keccak Reference. 2011. Available online: https://keccak.team/files/Keccak-reference-3.0.pdf (accessed 9th February 2021).
4. Arghya Bhattacharjee, Eik List, Cuauhtemoc Mancillas-López and Mridul Nandi. The Oribatida Family of Lightweight Authenticated Encryption Schemes. Submission to the NIST Lightweight Competition. September 27, 2019. Available online: https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/oribatida-spec-round2.pdf (Accessed February 9, 2021).
5. Avik Chakraborti, Nilanjan Datta, Ashwin Jha, Cuauhtemoc Mancillas-López, Mridul Nandi and Yu Sasaki. ESTATE: A Lightweight and Low Energy Authenticated Encryption Mode. IACR Transaction on Symetric Cryptology, Vol. 2020, Special Issue 1, pp. 350-389. 2020. doi:10.13154/tosc.v2020.iS1.350-389
6. Avik Chakraborti, Nilanjan Datta, Ashwin Jha, Cuauhtemoc Mancillas-López, Mridul Nandi and Yu Sasaki. INT-RUP Secure Lightweight Parallel AE Modes. IACR Transactions on Symmetric Cryptology, Vol. 2019, No. 4, pp. 81-118. 2019. doi:10.13154/tosc.v2019.i4.81-118
7. Avik Chakraborty and Mridul Nandi. Trivia-ck-v2. Second Round CAESAR Competition. August 28, 2015. Available online: https://competitions.cr.yp.to/round2/triviackv2.pdf (Accessed February 9, 2021).

8. Joan Daemen and Vincent Rijmen. The Design of Rijndael, The Advanced Encryption Standard (AES). Second Edition, Springer, 2020. doi: 10.1007/978-3-662-60769-5
9. Christoph Dobraunig, Maria Eichlseder, Florian Mendel, Martin Schläffer. Ascon v1.2. Final portfolio CAESAR Competition. September 15, 2016. Available online: https://competitions.cr.yp.to/round3/asconv12.pdf (Accessed February 9, 2021)
10. Christof Paar and Jan Pelzl. Understanding Cryptography: A Textbook for Students and Practitioners.
11. Springer, 2009. doi:10.1007/978-3-642-04101-3
12. Matthew Robshaw and Olivier Billet Editors. New Stream Cipher Designs, The eSTREAM Finalists. Springer, 2008. doi:10.1007/978-3-540-68351-3
13. Phillip Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. Advances in Cryptology - {ASIACRYPT} 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, December 5-9, 2004, Proceedings. Lecture Notes in Computer Science, Vol. 3329, pp. 16-31. Springer 2004.
14. Phillip Rogaway and Thomas Shrimpton. A Provable-Security Treatment of the Key-Wrap Problem. Advances in Cryptology – EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings. Lecture Notes in Computer Science, Vol. 4004, pp. 373-390, Springer, 2006. doi:10.1007/11761679_23
15. Kazuo Sakiyama, Yu Sasaki and Yang Li. Security of Block Ciphers: From Algorithm Design to Hardware Implementation. John Wiley & Sons. 2015. doi: 10.1002/9781118660027
16. Victor Shoup. A Computational Introduction to Number Theory and Algebra. Cambridge University Press. 2005. doi:10.1017/CBO9781139165464
17. Douglas R. Stinson and Maura B. Paterson. Cryptography Theory and Practice (4th edition). Chapman and Hall/CRC. 2018.
18. Hongjun Wu. ACORN: A Lightweight Authenticated Cipher (v3). Final portfolio CAESAR Competition. September 15, 2016. Available online:
19. https://competitions.cr.yp.to/round3/acornv3.pdf (Accessed February 9, 2021).

Tópicos Selectos en Criptografía

Objetivo

Presentar descubrimientos recientes en criptografía. Aprender a leer, analizar y estudiar artículos científicos recientes y relevantes en el área de criptografía de curvas elípticas y de emparejamientos bilineales.

Descripción

El curso inicia con un análisis y recuento de las primitivas usadas en criptografía, seguido por una discusión de los esquemas que han sido propuestos recientemente para realizar criptografía simétrica y de clave pública.
El curso está dirigido a estudiantes de Maestría y de Doctorado con interés especial en criptografía.

Contenido

1. Marco Teórico
a. Funciones de un solo sentido
b. Funciones y permutaciones seudo-aleatorias
c. Generadores seudo-aleatorios.

2. Criptografía de clave simétrica
a. Nociones y definiciones de seguridad
b. Modos de operación en cifradores por bloque
c. Códigos de autenticación de mensajes

3. Criptografía de la clave pública
a. RSA-OAEP
b. El problema del logaritmo discreto e hipótesis de Diffie Hellman

4. Criptografía de curvas elípticas
a. Preliminares matemáticos
b. Introducción a curvas elípticas
c. Aritmética de curvas elípticas
d. Curvas con endomorfismo que pueden calcularse eficientemente.

5. Criptografía basada en emparejamientos
a. Conceptos básicos
b. Protocolos
c. Emparejamiento de Tate.
d. Cómputo eficiente de emparejamiento de Tate

Bibliografía

a. Roberto M. Avanzi, Henri Cohen, Christophe Doche, Gerhard Frey, Tanja Lange, Kim Nguyen, Goldreich: Foundations of Cryptography. Cambridge University Press, 2004

b. Darrel Hankerson, Alfred Menezes, and Scott Vanstone. Guide to Elliptic Cryptography. Springer-Verlag, New York, 2004.
c. Michael George Luby: Pseudorandomness and Cryptographic Applications. Princeton University Press, 1996.
d. Menezes, P. van Oorschot, S. Vanstone: Handbook of Applied Cryptography, CRC Press, Boca Raton, 1996
e. D. Stinson: Cryptography-Theory and Practice, CRC Press, 2006
f. Frederik Vercauteren. Handbook of Elliptic and Hyperelliptic Curve Cryptography Champan & Hall/CRC, 2005.
g. Lawrence C. Washington Elliptic Curves: Number Theory and Cryptography, Second Edition (Discrete Mathematics and Its Applications). Chapman & Hall/CRC; 2 edition (April 3, 2008).

Computabilidad y complejidad

Objetivo

Conocer los diversos paradigmas de computación formal, entre estos las funciones recursivas y las máquinas de Turing y examinar las limitaciones de la noción de computabilidad y los criterios diversos de clasificación de problemas atendiendo a la complejidad de sus procedimientos de solución.

Descripción

Se presenta a las funciones recursivas siguiendo el enfoque de máquinas de Turing, de programas-while y el puramente formal. Luego se presenta los grados de irresolubilidad para pasar después las jerarquías temporal y espacial de los problemas tratables. Ya para terminar, presentaremos una colección de problemas completos-NP. Al último presentamos la noción de complejidad abstracta debida a Kolmogorov.

Contenido

1. Conceptos básicos
a. Pruebas matemáticas
b. Numerabilidad

2. Funciones computables
a. Lenguajes formales de programación
b. Máquinas de Turing y problema de la parada
c. Tesis de Church
d. Propiedades de cerradura de funciones computables
e. Funciones elementales
f. Jerarquía de Grzegorczyk
g. Función de Ackermann

3. Universalidad y Teoremas de Recursión
a. Funciones recursivas
b. Decidibilidad
c. Teoremas de autorreproducción
d. Virus en programas-while

4. Irresolubilidad
a. Teorema de Rice
b. Problema de la correspondencia de Post
c. Irresolubilidad de problemas en GLC
d. Algunos otros problemas irresolubles
e. Irresolubilidad en la Aritmética Aritmética de Peano
f. Clasificación de problemas irresolubles

5. Las Jerarquías de Computabilidad
a. Ordenes de funciones
b. Clases de problemas: Problemas de decisión, problemas de solución
c. Comprobadores y resolvedores

d. Complejidades de tiempo y espacio
e. Jerarquía en espacio
f. Jerarquía en tiempo

6. Problemas difíciles y completos en clases polinomiales
a. Algunos problemas principales completos-NP
b. Problemas de acotación
c. Problemas de acceso
d. Problema de acceso generalizado

7. Algoritmos probabilistas
a. Método de Monte-Carlo para probar primicidad
b. Máquinas probabilistas

8. Complejidad de Kolmogorov

Bibliografía clásica

a. Aho, Ullman: Foundations of computer science, W.H. Freeman & Co., 1992
b. Anderson, Turing et al: Mentes y máquinas, en la colección Problemas científicos y filosóficos, Universidad Nacional Autónoma de México, 1970
c. Arbib, Kfoury, Moll: A basis for theoretical computer science, Springer-Verlag, 1981
d. Balcazar. Diaz, Gabarro: Structural complexity I, Springer-Verlag, 1988
e. Davis: Computability and unsolvability, Mc-Graw Hill, 1958
f. Garey, Johnson: Computers and intractability: A guide to the theory of NP-completeness, Freeman, 1979

Bibliografía

1. Ganesh Gopalakrishnan, Automata and Computability: A Programmer's Perspective, Chapman and Hall/CRC; 1st edition, 2019
2. Gerard Prudhomme, Language Computability and Formal Language Theory, Society Publishing 2017
3. Rebecca Weber, Computability Theory, Orient Publication; 1st edition, 2016

Computación Paralela

Objetivo:
El alumno aprenderá los conceptos de la programación paralela, sus paradigmas, técnicas y estrategias de programación y evaluación.

Contenido:

1. Introducción a la computación paralela
    - 1.2 Definiciones básica
    - 1.3 Taxonomía de programas paralelos
    - 1.4 Arquitecturas paralelas
    - 1.4.1 Arquitecturas basadas en memoria
    - 1.4.2 Arquitecturas multicore y manycore
    - 1.4.3 Constelaciones
    - 1.4.4 Arquitecturas basadas en clusters
    - 1.4.5 Arquitecturas híbridas-heterogeneas
    - 1.5 Diseño de algoritmos paralelos
    - 1.5.1 Modelo tarea-canal
    - 1.5.2 Metodologia de Ian-Foster
    - 1.5.3 Patrones de programas paralelos

2. Programación Multicore
    - 2.1 Modelo de memoria compartida y tecnologías de desarrollo
    - 2.2 Hilos
    - 2.3 OpenMP
    - 2.4 Algoritmos SIMD y MIMD

3. Análisis de rendimiento de programas paralelos
    - 3.1 Modelo de tiempo de ejecución
    - 3.2 Aceleración y eficiencia
    - 3.3 Ley de Amdahl
    - 3.4 Ley de Gustafson-Barsis
    - 3.5 Métrica Karp-Flatt
    - 3.6 Métrica de isoeficiencia
    - 3.7 Análisis de consumo energético

4. Programación Many-Core
    - 4.1 Modelos de programas en arquitecturas heterogéneas
    - 4.2 Lenguajes y herramientas de desarrollo
    - 4.3 Programación en CUDA y OpenACC
    - 4.4 Programación para XEON-PHI

5. Programación para memoria distribuida
    - 5.1 Modelos de programación
    - 5.2 Programación en MPI
    - 5.2.1 Modelo de paso de mensaje
    - 5.2.2 La interfaz de paso de mensajes
    - 5.2.3 Funciones básicas y ejecución de programas MPI

5.2.4 Introducción a la comunicación colectiva
5.2.5 Medición de tiempos de ejecución.
5.2.6 Comunicación punto a punto.
5.2.7 Comunicación Avanzada (Scatter, Gather, Broadcast, All to All)
5.2.5 Ejemplos.

6. Programación multi-many core
6.1 Revisando la metodologia de Foster y patrones de programas paralelos.
6.2 Programación OpenMP-CUDA
6.2 Algoritmos y programas paralelos híbridos
6.3 Algoritmos y programas paralelos híbridos-heterogéneos

Bibliografía

1.-Quin, M. "parallel programming in C with MPI and OpenMP." McGraw Hills, edition (2007).
2.-Chandra, Rohit, et al. Parallel programming in OpenMP. Morgan kaufmann, 2001.
3.- Mattson, Timothy G., Beverly Sanders, and Berna Massingill. Patterns for parallel programming. Pearson Education, 2004.
4.- Sanders, Jason, and Edward Kandrot. CUDA by example: an introduction to general-purpose GPU programming. Addison-Wesley Professional, 2010.
5.- Soyata, Tolga. GPU parallel program development using CUDA. CRC Press, 2018.
6.- Morse, H. Stephen. Practical parallel computing. Academic Press, 2014.

Cómputo Móvil

Objetivo

El objetivo principal de este curso es el de proporcionar un marco general de los fundamentos, métodos y tecnologías para la compresión de problemáticas y el desarrollo de aplicaciones en el área de computación móvil. Adicionalmente a la formación teórica, el curso tendrá una fuerte orientación práctica al desarrollo de aplicaciones en dispositivos móviles, particularmente en teléfonos celulares con soporte para Android.

Descripción

La necesidad de información en cualquier momento y lugar, conjuntamente con el surgimiento de dispositivos de cómputo portátiles y los avances en las tecnologías de comunicación inalámbrica e Internet, han hecho a la Computación Móvil una realidad. Esta tiene como finalidad, el tratamiento automático de información por medio de dispositivos computacionales con capacidad de movilidad y con acceso digital a fuentes de información vía una infraestructura de comunicación inalámbrica. Los ambientes de cómputo móvil se caracterizan por restricciones importantes de recursos y cambios frecuentes en las condiciones de operación lo cual impone desafíos que involucran diversas áreas de las ciencias computacionales, ingenieríaś computacional, eléctrica y de telecomunicaciones.

Contenido

1. Introducción y motivación
2. Fundamentos de cómputo móvil
3. Tecnologías y plataformas móviles
4. Desarrollo de aplicaciones
5. Administración de recursos móviles
6. Localización
7. Redes y seguridad
8. Android

Bibliografía

a) B'Far, R., & Fielding, R. (2004). Mobile Computing Principles: Designing and Developing Mobile Applications with UML and XML. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511546969
b) Tommi Mikkonen, Programming Mobile Devices, John Wiley and Sons, 2007.
c) K. Pahlavan and Allen H. Levesque, Wireless Information Networks, Wiley-Interscience, Second edition, 2005.
d) Greg Nudelman, Android design patterns: Interaction design solutions for developers, Wiley, 1st Ed., 2013.
e) Karim Yaghmour, Embedded Android: Porting, Extending, and Customizing. O'Reilly Media, 1st, 2013.
f) Greg Milette, Adam Stroud, Professional Android Sensors Programming. Wrox, 1st Ed., 2012.
g) Erik Hellman, Android Programming: Pushing the limits. Wiler, 1st Ed., 2013.
h) IEEE Computer Magazine.
i) IEEE Transactions on Mobile Computing.

Cómputo Ubicuo

Objetivo

El objetivo de este curso es de introducir a los estudiantes en esta área de un sistema de investigación, la cual está cobrando cada vez más importancia, esto se puede apreciar gracias a la gran cantidad de congresos y publicaciones que se están produciendo constantemente

Descripción

El cómputo ubicuo, también conocido como cómputo pervasivo, es considerado la tercera ola de la computación. La visión del cómputo ubicuo es de integrarse en el ámbito físico, social y humano de los usuarios. La integración física para por el desarrollo de nuevos dispositivos que pueden ser utilizados en el ambiente físico la integración en el ambiente social y humano se da porque los nuevos dispositivos son ahora más accesibles y fáciles de usar por el usuario, además de ser diseñados para operar en armonía con el ambiente social.

Contenido

1. Introducción

a) Definiciones básicas de Computo Ubicuo
b) Antecedentes
c) ¿Por qué estudiar Cómputo Ubicuo?

2. Aplicaciones y requerimientos para la implementación de un sistema de Cómputo Ubicuo

a) Ejemplos de proyectos desarrollados
b) El cómputo ubicuo en la vida diaria.

3. Consideraciones para la implementación de sistemas en Cómputo Ubicuo
    a. Tópicos y retos para implementar sistemas en cómputo Ubicuo
    b. ¿Por qué crear sistemas en Cómputo Ubicuo?
    c. Diseño, implementación y documentación

4. Dispositivos móviles inteligentes y servicios inteligentes

b. Características de los dispositivos inteligentes
c. Ciclo de vida de los servicios inteligentes
d. Diseño de servicios móviles
e. Dispositivos y usuarios móviles
f. Elementos de red

5. Interacción en computo ubicuo

1. Diseño centrado en el usuario
2. Interacción implícita vs interacción explícita
3. Interacción oculta

6. Sistemas conscientes del contexto

a) Consciencia de contexto
b) Aplicaciones conscientes al contexto
c) Diseño e implementación sistemas conscientes al contexto
d) Conciencia temporal, especial y de movilidad
e) Consideraciones al implementar un sistema consciente del contexto

7. Inteligencia ambiental

a) Conceptos básicos
b) Arquitectura de sistemas inteligentes
c) Representación del conocimiento
d) Interacción inteligente
e) Multiplicidad de interacción
f) Ejemplos de implementaciones
g) Retos y perspectivas del cómputo ubicuo

Referencias

a. S. Posland, Ubiquitous Computing – smart Devices, Environmentsans Interactions, John Wiley and Sons, 2009, ISBN: 978-0-470-03560-3
b. J. krumm, Ubiquitous Computing Fundamentals, CRC Press, 2010, ISBN: 978-1-4200-9360-5
c. Q, Li and T., Shih, Ubiquitous Multimedia Computing, CRC Press, 2010, ISBN: 978-1-4200-9338-4
d. S. Kouadri, Advances in Ubiquitous Computing: Future Paradigms and Direction, IGI Piblishing, 2008, ISBN: 978-1-59904-842-0.
e. Ulrik Ekman, Jay David Bolter, Lily Diaz, Morten Sondergaard, and Maria Engberg. 2017. Ubiquitous Computing, Complexity, and Culture (1st. ed.). Routledge, USA.
f. Popkova, Elena G. Ubiquitous Computing and the Internet of Things: Prerequisites for the Development of ICT. 2019. <https://doi.org/10.1007/978-3-030-13397-9>.

Pruebas y Confiabilidad del Software

Objetivo

El curso de pruebas de confiabilidad de software permitirá al alumno comprender las técnicas necesarias para mejorar la calidad del software. Cualquier sistema de software con fallas en su funcionamiento o que se ejecute fuera de si especificación tendrá poca duración o será rechazado por el cliente.

Descripción

De acuerdo a estudios recientes en cientos de sistemas de software usados en los Estados Unidos, la gran mayoría de estos presentan numerosos defectos de software que perjudican la integridad de los sistemas que controlan o que perjudican a la economía o seguridad del personal que utiliza este software.

Es claro, que en la actualidad el software es utilizado para controlar una gran parte de la economía de los países altamente industrializados y se encuentran presentes en la mayoría de los aparatos electrónicos que utilizamos diariamente. Es por esto que los sistemas de software cuenten con una alta confiabilidad, que permita que el software este siempre disponible, libre de fallos catastróficos, seguro, íntegro y que se ejecute siempre de acuerdo a como fue especificado.

En este curso se presenta primero una introducción a la confiabilidad de software, la cual permitirá introducir técnicas para una programación confiable que ayude a la eliminación de defectos. Así mismo, en este curso se presentarán distintas técnicas para la implementación de pruebas de software, inspecciones y depuración. Finalmente se presentará el mecanismo de diseño por contrato como un mecanismo para reducir los fallos en el diseño del software.

Cada capítulo de este curso tendrá una introducción teórica, un ejemplo de implementación y distintos ejercicios prácticos que permitan al alumno implementar las técnicas aprendidas.

A pesar de ser un curso impartido dentro de la especialidad de Ingeniería de Software, no se requiere de haber llevado el curso de Introducción a la Ingeniería de Software

Contenido

a. INTRODUCCION A LAS PUEBRAS Y CONFIABILIDAD DEL SOFTWARE
    1. Introducción a las pruebas y confiabilidad del software.
        a. Pruebas de software como proceso de Ingeniería del Software.
        b. Confiabilidad
        c. Definiciones de error, fallo y fallo de sistema.
        d. Importancia de la detección de fallos en el software.

    2. Programación Modular: Primer paso hacia la programación confiable.
        a. introducción a la programación modular.
        b. Programación modular en la práctica
        c. Programación el lenguaje C estilo tradicional
        d. Mecanismos del lenguaje C para soporte del encapsulado.

    3. Metodología para la programación modular modelo de módulos.
        a. estructura de archivos asociada al modelo de módulos
        b. modelo de la interfaz y las dependencias de los módulos y sus archivos

c. estructura de los archivos del encabezado
d. directivas para el diseño de módulos

4. Ejemplos y actividad práctica.

b. FUNDAMENTOS DE CORRECCION Y CONFIABILIDAD

1. Conceptos básicos sobre defectos del software
2. Programación disciplinada
3. Tipos de datos abstractos
4. Diseño por contrato
5. Invariantes
6. Ventajas del uso consciente de aseveraciones
7. Estilo de codificación del software
8. Inspecciones, pruebas y depuración del software
9. Disciplinas de desarrollo
10. Ejemplos y actividad práctica

c. FUNDAMENTOS DE PRUEBAS DE SOFTWARE

1. Conceptos básicos sobre pruebas del software
2. Pruebas de Caja Negra
3. Pruebas estructurales o de Caja Blanca
4. Pruebas como mecanismo de evitación de defectos
5. Papel de las aseveraciones ejecutables durante las pruebas

d. AUTOMATIZACION DE LAS PRUEBAS

a. Proceso de Automatización de las pruebas de unidad
b. Requerimientos, Diseño e Implementación del Módulo de Pruebas
c. Ejemplos de Prueba del Módulo
d. Actividad Práctica

e. INSPECCIONES DE SOFTWARE

j) Inspecciones del software para la detección de defectos
k) Inspecciones formales
l) Técnicas de detección de defectos durante la inspección
m) Aspectos psicológicos de la inspección
n) Inspecciones formales como mecanismo de prevención de defectos
o) Otras prácticas de inspección menos formales
p) Conclusiones

f. FUNDAMENTOS DE DEPURACION DEL SOFTWARE

a) Conceptos básicos sobre depuración del software
b) observación de los hechos
c) Bitácora de la ejecución para la depuración
d) Trazas o historia de la ejecución

e) Verificación automática del estado del programa
f) Ejemplos y Actividad Práctica

g. FUNDAMENTOS DE DISEÑO POR CONTRATO

a. Diseño por contrato
b. Invariables.
c. Ventajas del uso consistente de aseveraciones
d. Aseveraciones vs. Pruebas de Unidad
e. Diseño de módulo para el soporte del diseño por contrato en lenguaje C
f. Ejemplos y Actividad Práctica

Diseño de Herramientas para el Desarrollo de Sistemas Distribuidos

Objetivo

Las aplicaciones distribuidas son cada vez más comunes. Las arquitecturas son variadas dependiendo del problema que se esté resolviendo. El objetivo de este curso es discutir la teoría y estrategias para el diseño de middlewares y sus respectivas API`s para este tipo de aplicaciones.

a. Introducción a sistemas distribuidos

b. Diseño de API´s
    a. Definiciones y Métricas
    b. Proceso de diseño
    c. Guías de diseño

c. Diseño de middlewares
    a. Introducción a middlewares
    b. Elementos de middlewares
    c. Arquitecturas
    d. Interoperabilidad
    e. Performance y Escalabilidad
    f. Manejo de sistemas
    g. Seguridad
    h. Diseño y arquitecturas IT

d. Patrones de diseño
    a. Análisis y diseño orientado a patrones
    b. Composición de patrones
    c. Patrones y UML
    d. Proceso de desarrollo usando patrones

e. Patrones de diseño para Sistemas Distribuidos
    a. Modelo de capas
    b. Patrones de infraestructura para distribución
    c. Manejo de eventos
    d. Patrones de infraestructura de concurrencia
    e. Patrones para sincronizar
    f. Adaptación y extensión
    g. Manejo de recursos

Bibliografía

1. Veríssimo, Paulo. "Distributed system for system architects"; Kluwer Academic, 2001.
2. Serain, Daniel. "Middleware", Springer-Verlag, 1999.
3. Tulach, Jaroslav; "Practical API Desing: Confessions of a Java Framework Arcitect"; Jaroslav Tulach; 2008.
4. Britton, Chris & Bye, Peter; "IT Architectures and Middleware"; Addison Wesley, 2004.
5. Puder, Arnor, Römer, Kay & Pilhofer, Frank; "Distributed Systems Architectures"; Morgan Kaufman, 2005.

6. B´Far Reza; ”Mobile Computing Principles: designing and developing mobile applications with UMLand XML”, Cambridge Press, 2009.
7. Sherif M. Hany, Ammar, Pattern-Oriented Analysis and Design: Composing Patterns to Design Software Systems, Addison Wesley, 2003.
8. Bushman F., Henney, K. D. Schmidt, Patter-Oriented Software Architecture, A pattern for Distributed Computing V. 4, Wiley and Sons, 2007.

Tópicos selectos de teoría de juegos

Objetivo

Estudio de conceptos, modelos, algoritmos y métodos en aprendizaje automático y teoría de juegos, para analizar y simular estrategias de reconocimiento de patrones en procesos biológicos y médicos.

Descripción

En el curso estudiamos métodos de aprendizaje automático, clásico y profundo, y de razonamiento y aprendizaje de estrategias en teoría de juegos (TJ). Su implementación es con redes neuronales artificiales, clásicas o profundas. Su aplicación en el reconocimiento de patrones emergentes en biología o medicina. En TJ, un juego es un modelo matemático sobre la colaboración o competencia entre jugadores para lograr un objetivo. El modelo integra las reglas y condiciones del juego, así como las acciones y estrategias entre jugadores y equipos. El equilibrio de Nash habilita caracterizar los estados durante la evolución de diversos juegos. Estudiamos como aplicar este equilibrio para elegir estrategias en juegos como béisbol y futbol soccer o americano. El juego de Go, de tablero y reglas simples pero crecimiento combinatorio exponencial, es representativo de problemas actuales muy relevantes en computación, medicina y biología. Las interacciones entre (gran cantidad) de elementos simples a partir de las cuales emergen patrones de comportamiento complejos es el punto de relevancia. El modelado y simulación de tales patrones y sus distribuciones de probabilidad se estudian en el curso. Los conceptos y métodos en aprendizaje automático y TJ se aplican en el desarrollo de algoritmos para el reconocimiento de patrones como metástasis en cáncer y la reacción del sistema inmune; y en el diseño de estrategias para aumentar la probabilidad de éxito en un juego. La bioinformática y la informática médica son áreas de creciente demanda, muy alta. En parte, por el reconocimiento de patrones y comportamientos emergentes, a diversas escalas, lo cual se logra con métodos y algoritmos de aprendizaje automático, TJ y ciencia de datos. Por eso su importancia.

Contenido

1. Teoría de Juegos: estrategias, multi-jugador, competitivos y cooperativos. El concepto de equilibrio.
    - A. Equilibrio de Nash para elegir estrategias en juegos de equipo.
        - a. Baseball, Futbol Americano, Soccer, Ajedrez.
        - e. Equivalencia formal entre estos juegos.
    - B. Lenguajes formales
        - a. Reglas de juego y su gramática. Autómatas de estados finitos.
        - b. Juego en forma normal:
            - i. Tácticas y estrategias.
            - ii. Función de costo – beneficio.
    - C. Distribución de probabilidad: Ley de potencia.
2. Aprendizaje automático
    - A. Redes neuronales, naturales y artificiales.
    - B. Aprendizaje clásico supervisado y no supervisado.
    - C. Aprendizaje profundo (Dr. Didier Barradas Bautista):
        - a. Redes neuronales profundas y convolucionales.
        - b. Máquinas restringidas de Boltzman (RBM).
    - D. Interacción estocástica:
        - a. Modelo de Ising.
    - b. Patrones de convergencia.

3. Aplicaciones en bioinformática e informática médica
    A. El juego de Go, AlphaGoZero.
    B. Biología de sistemas.
        a. Metástasis en cáncer y la respuesta inmune.
        b. Patrones epidemiológicos.
    C. Redes complejas:
        a. centralidad, grado de un node y jerarquía.
        b. Redes de Mundo Pequeño y de libre escala.

Referencias

a. J. von Neumann and O. Morgenstern, Theory of Games and Economic Behavior, Princeton University Press, 2004.
b. Nash, J. F. Non cooperative games. The annals of mathematics, 2nd Series, 54 (2), USA, 1951.
c. Lewis, H. & Papadimitrou Ch., Elements of the theory of computation, 2nd edition Prentice-Hall, 1997.
d. S. Boccaletti, V. Latora, Y. Moreno, et al. Complex networks: Structure and dynamics. Elsevier, 2005, Vol. Physics Reports. 0370-1573.
e. Rojas Raul, Neural Networks: a systematic study. Springer Verlag, 2004.
f. D. Silver, A. Guang, A. Guez, et al, Mastering the game of Go with deep neural networks and tree search, Nature, 529, 42016.
g. M.E.J. Newman. Power Law, Pareto Distributions and Zips's Law. In Contemporary Physics, 46:5, 323-351, 2005. https://doi.org/10.1080/00107510500052444.
h. Chris Wilson, Game Theory: A Guide to Game Theory, Strategy, Economics and Success!, Ingram Publishing, 2020, ISBN: 978-1761032554.

Lecturas

i. Alvarado Matías, Yee Arturo & Cocho Germinal, Simulation of baseball gaming by cooperation and non-cooperation strategies. Computación y Sistemas, 2014, 18 (4), 693- 708. doi: 10.13053/CyS-18-4-1987
j. Yee A., Rodriguez R, Alvarado M. Analysis of strategies in American Football using Nash equilibrium. In Artificial Intelligence Methods Systems and Applications, AIMSA 2014. Doi: https://link.springer.com/chapter/10.1007/978-3-319-10554-3_30
k. Alvarado M. and Yee, A. Nash equilibrium for collective strategic reasoning. Expert Systems with Applications 39 (15) 1014 – 1025, Elsevier Science 2012, doi: https://www.sciencedirect.com/science/article/abs/pii/S0957417412005738?via%3Dihub
l. Tellez Giron J. and Alvarado M. Computer football: Plays, Players and Strategies Choice. IEEE Latin America Transactions 16 (5) 2018. DOI: https://ieeexplore.ieee.org/document/8408445
m. Rojas A., Barradas Didier and Alvarado, M., Modeling the game of Go by Ising Hamiltonian, Deep Belief Networks and Common Fate Graphs. In IEEE Access 2019. DOI: https://ieeexplore.ieee.org/document/8717638
n. Barradas Didier, Alvarado Matías, Agostion M, Cocho Germinal. Cancer growth and metastasis as a metaphor of Go gaming: An Ising model approach. I

Geometría Computacional

Descripción

En este curso se estudiarán los conceptos básicos de la Geometría Computacional (GC) así como sus aplicaciones. El material aborda las técnicas necesarias para el diseño y análisis de algoritmos eficientes para resolver problemas en geometría, tales como: cubiertas convexas (*convex hulls*), intersecciones geométricas, diagramas de Voronoi, triangulaciones de Delaunay, estructuras de datos geométricas, etc.

Contenido

1. Introducción
    1. Representación de objetos geométricos elementales: puntos, segmentos, polígonos.
    2. Operación elemental: vuelta a la izquierda y a la derecha. Aplicaciones de la misma.
2. Cierres convexos
    1. Definición y caracterización
    2. Dos algoritmos de fuerza bruta: encontrar puntos extremos
    3. Algorithms:
        a) Jarvis march
        b) Quick Hull (prune and search)
        c) Graham scan
        d) Algoritmo incremental
        e) Divide y vencerás
        f) Cota inferior
3. Intersección de segmentos de recta
4. DCEL: Doubly-Connected Edge List
5. Intersección de segmentos de recta
6. Triangulation de polygons
    1. Problema de la galería de arte
    2. Todo polígono admite una triangulación
    3. Propiedades de la triangulación de un polígono
    4. Algoritmos para triangular un polígono:
        a) interstar diagonales,
        b) remover orejas,
        c) polígono convexo y polígono monótono;
        d) partición en piezas monónonas.
7. Búsquedas en rangos ortogonales
    1. Rangos 1-dimencionales
    2. Kd-trees
8. Problemas de proximidad
    1. Diagrama de Voronoi: Definición, caracterización y propiedades
    2. Cómo se almacena el DV
    3. Algoritmos para construir el DV:
        a) Fuerza bruta b) Incremental c) Divide y vencerás
    4. Solución de problemas de proximidad usando el DV

Bibliografía

1.-Mark de Berg, Otfriend Cheong, Marc van Kreveld, Mark Overmars. 2008. Computational Geometry: Algorithms and Applications (Third ed.). Springer.
2.-Jouseph O'Rourke. 1998. Computational Geometry in C (Second ed). Cambridge University Press.

Graficación

Objetivo

El objetivo del curso es estudiar las diversas técnicas de la literatura para el trazado de una escena y de las formas bi- y/o tridimensionales que la componen, además de las técnicas para manipularla y visualizarla. La herramienta sugerida de trabajo, para desarrollar las tareas del curso, es un sistema de desarrollo de interfaces gráficas (GUI, por sus siglas en inglés) basado en objetos, llamado Qt (www.trolltech.com) y OpenGL (www.opengl.org) o Mesa (www.mesa3d.org) para la construcción y manipulación de escenas tridimensionales.

Contenido

a) Introducción.
   1) Definición y temas que estudia graficación.
   2) Arquitectura para despliegue tipo raster
   3) Marco de trabajo conceptual

b) Algoritmos básicos para trazo de primitivas en 2D

a. Trazo líneas con el algoritmo incremental de punto medio
b. Trazo de círculos con el algoritmo incremental de punto medio
c. Rellenado de polígonos.
d. Trazado de fractales con el lenguaje LOGO
e. Primitivas gruesas: líneas, círculos y polígonos.
f. Cortado (clipping).

c) Transformaciones geométricas

1. Translación, escalamiento, rotación y sesgado (sheared)
2. Las transformaciones en coordenadas homogéneas.
3. Composición de transformaciones 2D
4. Representación matricial de transformaciones 3D
5. Composición de transformaciones 3D
6. Las transformaciones como un cambio en el sistema de coordenadas.

d) Visión en 3D

1. Proyecciones: paralelas, en perspectiva.
2. Especificación de una vista arbitraria.
3. Deducción de ecuaciones de las proyecciones geométricas planas.

e) Representación de curvas y superficies
   a) Mallas de polígonos
   b) Curvas cúbicas paramétricas: Hermite, Bézier y B-splines.
   c) Superficies cúbicas paramétricas
   d) Superficies cuádricas.

f) Determinación de la superficie visible

1. Funciones de dos variables
2. El algoritmo de buffer z

g) Modelado de sólidos

1. Operaciones Booleanas regularizadas
2. Instanciamiento de Primitivas
3. Representaciones de barrido
4. Representaciones que particionan el espacio: Descomposición de celdas, enumeración de ocupancia espacial y octrees.
5. Geometría sólida constructiva.

h) Iluminación y sombreado

1. Modelos de iluminación
2. Modelos de sombreado para polígonos.
3. Sombras

i) Dibujado por trazo de rayo

a. Algoritmo básico para el trazo de rayos
b. Cálculo de las intersecciones rayo-superficie

Bibliografía

1. John Hughes, Andries Van Dam, Morgan McGuire, David Sklar and James Foley, Computer Graphics: Principles and Practice, 3rd edition, Addison Wesley, 2013, ISBN: 978-0321399526.
2. John Kessenich, Graham Sellers and Dave Shreiner, OpenGL Programming Guide: The Official Guide to Learning OpenGL Version 4.5 with Spri-V, Addison-Wesley, 2016, ISBN: 978-0134495491.
3. Gabriel Gambetta, Computer Graphics from Scratch: A Programmer's Introduction to 3D Rendering, No Starch Press, 2021, ISBN: 978-1718500761.
4. V. Scott Gordon and John Clevenger, Computer Graphics Programming in OpenGl with C++, Second Edition, Mercury Learning and Information, 2020.

Mineria de datos

Course description

The quantity and variety of online data is increasing very rapidly. The data mining process includes data selection and cleaning, machine learning techniques to “learn” knowledge that is “hidden” in data, and the reporting and visualization of the resulting knowledge. This course will cover these issues and will illustrate the whole process by examples of practical applications from the life sciences, computer science, and commerce. Several machine learning topics including classification, prediction, and clustering will be covered.

Contents

a. To introduce basic applications, concepts, and techniques of data mining, including
b. data and data pre-preprocessing
c. classification,
d. machine learning and deep learning,
e. association analysis, o cluster analysis,
f. anomaly detection, o other issues of data analysis.
g. To develop basic skills of data science for solving practical problems in a variety of disciplines.
h. To train students’ independent study ability and research thinking.
i. To practice data analysis and machine learning software, such as Weka, RapidMiner, R, Matlab,

Textbook

1.- Introduction to Data Mining (2nd edition), by Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, and Vipin Kumar, Pearson, 2018.

Ingeniería de software

Objetivo

Este curso tiene como objetivo que los estudiantes adquieran los conocimientos y competencias en el área de Ingeniería de Software necesarios para llevar a cabo exitosamente proyectos complejos de desarrollo de software.

Contenido

1. Introducción a la Ingeniería de Software
   a. Fallas en ingeniería de software
   b. ¿Qué es la ingeniería de software?
   c. Conceptos básicos de ingeniería de software
   d. Actividades de la ingeniería de desarrollo de software
   e. La gestión del desarrollo de software
   f. Los costos de la ingeniería de software
   g. Desafíos clave que enfrenta la ingeniería del software
   h. Ética profesional y responsabilidad

2. Procesos de software
   a) El proceso de software
   b) Modelo de proceso de Software
   c) El modelo de cascada
   d) Modelos incrementales de proceso
   e) Modelos evolutivos de proceso
   f) El Proceso Unificado (RUP)
   g) Modelos ágiles de procesos

3. Modelado con UML
   a) Conceptos básicos de modelado
   b) Diagramas de caso de uso
   c) Diagramas de clases
   d) Diagramas de interacción
   e) Diagramas de estado
   f) Diagramas de actividades

4. Ingeniería de requerimientos
   a. Obtención de requerimientos
   b. Análisis de requerimientos
   c. Validación de requerimientos
   d. Gestión de requerimientos

5. Modelado de análisis
   a. Análisis de requerimientos
   b. Conceptos y enfoques de modelado de análisis
   c. Modelado de datos
   d. Conceptos de modelado orientado a objetos
   e. Modelado basado en escenarios

f. Modelado basado en clases

6. Ingeniería de diseño
   a) Proceso de diseño
   b) Conceptos de diseño
   c) Modelo de diseño
   d) Arquitectura del software
   e) Diseño de datos
   f) Arquitecturas y patrones
   g) Diseño de arquitectura
   h) Diseño de la interfaz de usuario

7. Verificación y validación del software
1. Estrategias de prueba para software tradicional
2. Estrategias de prueba para software orientado a objetos
3. Pruebas de validación
4. Pruebas de sistema
5. Depuración
6. Fundamentos de pruebas de software
7. Pruebas de caja blanca
8. Pruebas de ruta básica de ejecución
9. Pruebas de estructura de control
10. Pruebas de caja negra
11. Métodos de prueba orientados a objetos

8. Métricas para el software
   a. Calidad del software
   b. Métrica para el modelo de análisis
   c. Métricas para el modelo de diseño
   d. Métricas para código fuente
   e. Métricas para pruebas de software
   f. Métricas para el mantenimiento

Bibliografía

- Roger S Pressman. Software Engineering: A Practitioner's Approach, 2004, 6th edition, McGraw-Hill; ISBN-10: 007301933X
- Ian Sommerville. Software Engineering, 2006, 8th Edition, Addison Wesley; ISBN-10: 0321313798
- Kent Beck. Extreme Programming Explained: Embrace Change, 2000, Addison-Wesley; ISBN-10: 0201616416
- Bernd Bruegge and Allen H. Dutoit. Object-Oriented Software Engineering: Using UML, Patterns and Java, 2003, 2nd edition, Prentice Hall; ISBN-10: 0130471100

Inteligencia Artificial

Objetivo

Qué el alumno aprenda cual es el fundamento de la Inteligencia Artificial y adquiera la habilidad de resolver problemas dentro del campo de la IA, se propone proyectos para que el alumno ponga en práctica su habilidad en la implementación de algoritmos y creatividad en el tema de heurística con algún lenguaje de programación, de acuerdo con los proyectos de interés.

Contenido

1. Introducción a la inteligencia Artificial
    - 1.1. ¿Qué es la Inteligencia Artificial?
    - 1.2. ¿Qué es la Inteligencia Humana?

2. Panorama general de la Inteligencia Artificial: IA simbólica e IA numérica
    - 2.1. La evolución de la Inteligencia Artificial
    - 2.2. Líneas de investigación de la IA
    - 2.3. Resolución de problemas de la IA
    - 2.4. La IA desde el marco de las ciencias cognitiva

3. Resolución de problemas y Heurística.
    - 3.1. ¿Qué es la Resolución de problemas?: How to solve it?
    - 3.2. El General Problem Solver y el inicio de la IA
    - 3.3. ¿Qué es Heurística?

4. Representación de conocimiento
    - 4.1. El problema de Representación y Significado de Ciencias de la Computación.
    - 4.2. Representación en el modelo semántico de base de datos
    - 4.3. Representación en el modelo de clases
    - 4.4. Representación en script (lenguaje natural)
    - 4.5. Representación en el modelo de marcos (frames) y orientado a objetos
    - 4.6. Representación en sistema de reglas de producción
    - 4.7. Transformación en lenguaje objeto de scheme

5. Conocimiento, Razonamiento y Planeación
    - 5.1. Lógica de primer orden
    - 5.2. Inferencia en lógica de primer orden
    - 5.3. Lógica de agentes
    - 5.4. Planeación clásica
    - 5.5. Conocimiento con incertidumbre

6. Métodos de Resolución de Problemas
    - 6.1. Esquemas para la Resolución de Problemas (RP)
    - 6.2. Espacio de conocimientos para la RP
    - 6.3. Métodos de la Heurística y combinatoria
    - 6.4. Algoritmos
    - 6.5. Síntesis de técnicas de optimización

7. Tópicos especiales de IA
    7.1. Aprendizaje
    7.2. Modelos probabilísticos
    7.3. Tópicos en Neurociencias

Bibliografía

Material en línea: http://e-cinvestav.cinvestav.mx/e-hypatia/elearning/

1. Panorama de la Inteligencia Artificial
2. Representación de conocimiento y razonamiento
3. Resolución del conocimiento: Heurística y Algoritmos
4. Introducción a la Combinatoria

Notas de clase:
1. Introducción a la lógica
2. Lógica y base de datos
3. Lógica: Demostración de teoremas y programación lógica

Introducción a la computación evolutiva

Objetivo

En este curso se estudiarán los conceptos básicos de las técnicas más importantes de computación evolutiva, haciendo especial énfasis en los algoritmos genéticos. Inicialmente, se harán un recorrido histórico en el que se resumirán los logros más importantes en torno a la simulación de los procesos evolutivos como una herramienta para el aprendizaje y la optimización. Posteriormente, se analizarán y compararan de manera general los 3 paradigmas principales que se utilizan hoy en día en la computación evolutiva: las estrategias evolutivas, la programación evolutiva y los algoritmos genéticos. En cada caso se abordará su inspiración biológica, su motivación, su funcionamiento y algunas de sus aplicaciones. Finalmente, se estudiará a mayor detalle el funcionamiento, fundamentos teóricos, implementación y operación de los algoritmos genéticos, que es actualmente el paradigma evolutivo más utilizado por los investigadores que trabajan en esta disciplina.

Contenido:

- Técnicas heurísticas
    a) Problemas P y NP
    b) Tecnicas clasicas de busqueda y optimización
    c) Lo que el mundo real demanda
    d) ¿Que es una heurıstica??
    e) ¿Realmente necesitamos tecnicas heurısticas??
    f) Ejemplos de tecnicas heurısticass
    g) Busqueda tabú
    h) Recocido simulado
    i) Escalando la colina
- Nociones de optimización
    a. Optimización Global
    b. Optimización Numérica
    c. Optimización Combinatoria
    d. Espacios de busqueda convexos y concavos
    e. Restricciones explıcitas se implıcitass
    f. Restricciones de igualdad y desigualdad
    g. Zona factible y no factible
- Antecedentes históricos
        a) Inspiración biológica
        b) Primeros intentos
        c) Cronología de descubrimientos importantes
- Paradigmas principales
    a. Estrategias evolutivas
    b. Programacion evolutiva
    c. Algoritmo genetico
    d. Comparaciones
- La computación evolutiva en el contexto de la Inteligencia Artificial

1. Criticas (IA Clásica vs. Técnicas Heurísticas))
2. ¿Un nuevo paradigma?

- Terminología biológica vs. terminología usada en la computación evolutiva
    a. Glosario Básico

b. Comparaciones

• Algoritmos Genéticos
   a. Generalidades
   b. Definición
   c. Componentes básicos
   d. Funcionamiento

• Representación
1. Binaria
2. Códigos de Gray
3. Real
4. Programacion Genética
5. Algoritmos genéticos desordenados (Messy GAs)
6. Otras propuestas

• Selección
1. Proporcional
2. Torneo
3. Estado Uniforme
4. Uso de jerarquıas

• Cruza
   a. Importancia
   b. Un punto
   c. Dos puntos
   d. Uniforme
   e. Casos especiales

• Mutación
   a) Importancia
   b) Forma básica
   c) No uniforme
   d) Casos especiales

• Función de aptitud
   a. Definición
   b. Uso de “cajas negras”
   c. Manejo de restricciones usando funciones de penalización´
   d. Ejemplos
   e.

• Ajuste de parámetros
1. Estudios empıricos
2. Auto-adaptación
3.

• Implementación
   Software propio
   Software de dominio publico
   Sistemas comerciales

• Operadores avanzados
1. Diploides y dominancia
2. Inversión
3. Micro operadores
4. · Segregación
5. · Traslocación
6. · Duplicación
7. · Borrado

• Teoría
1. Teorema de los esquemas
2. Modelos exactos
3. Teoría de convergencia
4. No Free Lunch Theorems
5. Críticas
6. Decepción

• ¿Cuando aplicar un algoritmo genético?
- Limitaciones
- Ventajas
- Áreas de aplicación

• Áreas abiertas de investigación
a. Inspiración biológica
b. Paralelismo
c. Teoría
d. Representación
e. Operadores
f. Coevolución
g. Algoritmos culturales
h. Ajuste de parámetros
i. Autoadaptación
j. Otras

• Temas avanzados (opcionales)
a) Funciones multimodales
b) · Nichos
c) · Repartición de aptitud
d) · Ejemplos
e) Funciones con objetivos múltiples
f) · Técnicas básicas
g) Manejo de restricciones
h) · Función de penalización
i) · Pena de muerte
j) · Separación de objetivos y restricciones
k) · Uso de representaciones y operadores especiales
l) · Otras propuestas

Bibliografía

a. A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing, Springer, Berlin, Second Edition, 2015, ISBN 978-3-662-44873-1.

b. David E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.

c. David Corne, Marco Dorigo & Fred Glover (editores), New Ideas in Optimization, McGraw-Hill, London, 1999.

d. Sadiq M. Sait & Habib Youssef, Iterative Computer Algorithms with Applications in Engineering, IEEE Computer Society, Los Alamitos, California, 1999.

e. Melanie Mitchell, An Introduction to Genetic Algorithms, MIT Press, Cambridge, Massachusetts, 1996.

f. Zbigniew Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, Second Edition, 1992.

g. John H. Holland, Adaptation in Natural and Artificial Systems. An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence, MIT Press, Cambridge, Massachusetts, Second Edition, 1992.

h. David B. Fogel, Evolutionary Computation. Toward a New Philosophy of Machine Intelligence, The Institute of Electrical and Electronic Engineers, New York, 1995.

i. Thomas Bäck, ¨Evolutionary Algorithms in Theory and Practice, Oxford University Press, New York, 1996.

j. David B. Fogel, Evolutionary Computation: The Fossil Record, The Institute of Electrical and Electronic Engineers, New York, 1998.

k. John R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, Massachusetts, 1992.

l. Colin B. Reeves (editor), Modern Heuristic Techniques for Combinatorial Problems, John Wiley & Sons, Great Britain, 1993.

m. Zbigniew Michalewicz & David B. Fogel, How to Solve It: Modern Heuristics, Springer, Berlin, 2000.

n. Hasmat Malik, Atif Iqbal, Puneet Joshi, Sanjay Agrawal, Farhad Ilahi Bakhsh, Metaheuristic and Evolutionary Computation: Algorithms and Applications, Springer, 2020.

Lenguajes de Programación

Objetivo

Estudiaremos los principales temas relacionados con el diseño y la implantación de los lenguajes de programación más representativos de los 4 principales paradigmas existentes en la actualidad: imperativo, funcional, orientado a objetos y lógico. Se estudiará la evolución de las estructuras de datos y de control contenidas en los lenguajes de programación, la motivación para su desarrollo y los compromisos que los diseñadores han tenido que considerar. Veremos cómo la fuerza principal que ha conducido muchas de las decisiones de diseño adoptadas en los lenguajes de programación han sido la búsqueda de una mejor ergonomía y el deseo de incrementar la productividad y confiabilidad en la producción de software. Además, se estudiarán diversos métodos para especificar formalmente la sintaxis de los lenguajes de programación, y los usaremos para ilustrar los compromisos existentes entre facilidad de procesamiento (de una computadora) contra legibilidad (de parte de un humano).

Contenido

a. La era del oscurantismo
    a. Lenguaje máquina
    b. Pseudo-código
    c. Ensambladores

2.- Primera generación: FORTRAN
- 2.1 Historia y motivación
- 2.2 Estructuras de datos y control
- 2.3 Aportaciones a otros lenguajes
- 2.4 Sintaxis
- 2.5 Paso de parámetros por referencia
- 2.6. Los grandes defectos del lenguaje
- 2.7 El debate acerca de su retiro

3.- Segunda generación: ALGOL-60
- 3.1. Historia y motivación
- 3.2. Estructuras de datos y control
- 3.3. Paso de parámetros por nombre
- 3.4. Las grandes aportaciones a los lenguajes modernos
- 3.5. Herramientas descriptivas: BNF (Backus-Naur Form)
- 3.6. Aspectos sintácticos
- 3.7. Los diagramas de contorno

4.- Tercera generación: Pascal
- 4.1. Historia y motivación
- 4.2. Estructuras de datos y control
- 4.3. Paso de parámetros por valor resultado
- 4.4. Los problemas con los arreglos y las cadenas
- 4.5. Aspectos sintácticos
- 4.6. Apuntadores
- 4.7. Estructuras heterogéneas
- 4.8. Enumeraciones

5.- Cuarta generación: Ada

5.1. Historia y motivación
5.2. Estructuras de datos y control
5.3. Paso de parámetros a la carta
5.4. Aspectos sintácticos
5.5. Paquetes
5.6. Modularidad
5.7. Concurrencia
5.8. El mecanismo de comunicaciones *Rendez-Vous*
5.9. Manejo de excepciones
5.10 Tareas
5.11 Sobrecarga de operadores
5.12 Paquetes genéricos y su comparación con las clases
5.13 ¿Da Ada las mejores soluciones a todo?
5.14 Las críticas al mejor lenguaje procedural de la actualidad

6.- Programación orientada a objetos: Smalltalk y C++
6.1 Historia y motivación
6.2. Clases, objetos, jerarquías, polimorfismo
6.3. Envío de mensajes
6.4. Reglas de ámbito estáticas y dinámicas
6.5. Las impurezas del C++
6.6. ¿Es realmente importante la programación orientada a objetos?

7.- Programación funcional: LISP, Scheme y ML
7.1. Historia y motivación
7.2. Sintaxis
7.3. Estructuras de control
7.4. Recursividad
7.5. Tipos
7.6. ¿Por qué importa la programación funcional?

8.- Programación orientada a la lógica: PROLOG
8.1. Historia y motivación
8.2. Aspectos sintácticos
8.3. Estructuras de control
8.4. Perspectivas y uso en Inteligencia artificial

9.- Introducción al Cálculo Lambda (opcional)
9.1. Igualdad de los términos lambda puros
9.2. Reglas de sustitución
9.3. Computación con términos lambda puros
9.4 Cálculo lambda con tipos
9.5. Polimorfismo

Bibliografía

1. Bruce J. MacLennan, *Principles of Programming Languages: Design, Evaluation, and Implementation*, Third Edition, Oxford University Press, ISBN 0195113063, March 1999.

2. D.E. Stevenson, *Programming Language Fundamentals by Example*, Auerbach Publications, Boca Raton, Florida, USA, 2007.

3. Maurizio Gabbrielli and Simone Martini, *Programming Languages: Principles and Paradigms*, Springer, London, UK, 2010.

4. Ravi Sethi & Tom Stone, *Programming Languages. Concepts and Structures*, Addison-Wesley Publishing Company, Second Edition, 1996.

5. Henri E. Bal & Dick Grune, *Programming Language Essentials*, Addison-Wesley Publishing Company, 1994.

6. Carlo Ghezzi & Mehdi Jazayeri, *Programming Languages Concepts,* John Wiley & Sons, Third Edition, 1997.

7. David Gelernter & Suresh Janagannathan, *Programming Linguistics,* MIT Press, 1980.

8. George Springer & Daniel P. Friedman, *Scheme and the Art of Programming*, The MIT Press, 1990.

9. Harold Abelson & Gerald Jay Sussman, *Structure and Interpretation of Computer Programs*, The MIT Press, 1985.

10. Rajiv Chopra, Principles of Programming Languages (POPL), I K International Publishing House, 2014.

11. Therese Hardin, Mathieu Jaume, Francoise Pessaux, Veronique Viguie Donzeau-Gouge, *Concepts and Semantics of Programming Languanges 2: Modular and Object-Orientes Constructs in Ocaml, Python, C++, Ada and Java*, Wiley, 2021.

Lógica y Bases de Datos

Objetivo

Proporcionar los elementos de diseño, optimización y explotación de bases de datos empleando técnicas de razonamiento automático.

Descripción

Las base de datos y la lógica se interrelacionan como sistemas deductivos en la solución de problemas visto como consultas en la base de datos. Los primeros temas consisten en una revisión de la lógica de predicados de primer orden y la demostración automática de teoremas para, posteriormente, encontrar una fundamentación para las bases de datos en el lenguaje relacional, restricciones de integridad y diseño de base de datos. La segunda parte es usar la aproximación de la lógica como sistema deductivo en las bases de datos.

Contenido

1. Introducción a la lógica matemática
a. Lógica proposicional
b. Lógica de predicados de primer orden
c. Demostración automática de teoremas

2. La lógica como fundamento a base de datos
a. El modelo relacional y lógica de predicados de primer orden
b. Restricciones de identidad y lógica
c. Suposición de un mundo cerrado en base de datos
d. Lógica para descripción de datos

3. Fundamentación lógica en el modelo entidad-relación
a. Estructura jerárquica de la lógica de predicados
b. Restricciones estáticas
c. Dinámica de las bases de datos
d. Generalización de las restricciones relacionales
e) Teoría de normalización de base de datos

4. Base de datos deductivas (BDD)
a. BDD y bases de datos lógicas
b. BDD definidas
c. BDD indefinidas

5. Sistemas de base de datos deductivas
a. Sistema R: aproximación relacional a bases de datos
b. Sistema CORAL
c. Estrategias para procesamiento de consultas recursivas

6. Temas selectos en Base de Datos
a. Lógica no monotonía
b. Suposición del mundo cerrado vs negación bajo falla

c. Retículas y unificación

Bibliografía básica

1. S. V. Chapa Vergara; “Lógica clásica”; Departamento de Ingeniería Eléctrica, CINVESTAV, México, 2000.
2. S. V. Chapa Vergara; “Lógica e inteligencia artificial” Departamento de Ingeniería Eléctrica, CINVESTAV, México, 2000.
3. S. V. Chapa Vergara; “Lógica y base de datos” Departamento de Ingeniería Eléctrica, CINVESTAV, México, 2000.
4. Minker J.; “Foundations of Deductive Databases and Logic Programming”, Morgan Kaufmann, Los Altos, CA, 1988.

Bibliografía

1. C.J. Date, Database in Depth: Relational Theory for Practitioners 1st Edition, O'Reilly Media, 2005
2. John Garmany, Jeff Walker, Terry Clark, Logical Database Design Principles (Foundations of Database Design) 1st Edition, Auerbach Publications; 1st edition, 2005
3. Toby Teorey, Sam Lightstone, Tom Nadeau, H.V. Jagadish, Database Modeling and Design
4. Logical Design, The Morgan Kaufmann Series in Data Management Systems, Fifth Edition, Elsevier Inc., 2011

Matemáticas discretas

Objetivo

Ofrecer al estudiante un panorama general de las Matemáticas que son particularmente útiles a las Ciencias de la Computación. Se inicia presentando las ideas básicas del principio de conteo y el razonamiento combinatorio elemental. A continuación, se ofrece una introducción general a la lógica matemática, un estudio riguroso de la teoría de conjuntos, el principio de la inducción matemática y los métodos recursivos. Posteriormente se estudian las relaciones y funciones y se termina con lenguajes y máquinas de estados finitos. El curso no supone conocimientos matemáticos profundos previos y se enfoca principalmente a desarrollar la capacidad del estudiante para resolver problemas.

Contenido

a. Relaciones binarias y gráficas.
   a) Relaciones binarias.
   1) Relaciones y gráficas.
   2) Interpretación matricial.
   b) Clases de relaciones.
   1) Relaciones de equivalencia.
   2) Relaciones de orden total y parcial.
   3) Retículas y conjuntos parcialmente ordenados.
   4) Orden parcial de relaciones de equivalencia.
   c) Gráficas y sus aplicaciones.
   d) Gráficas de subconjuntos.
   e) Gráficas de De Bruijn.
   f) Computación de gráficas con LGraph.
   g) Aplicaciones a ingeniería de software: análisis de datos, dependencias funcionales, etc.

b. Semigrupos, monoides e ideales.
a. Propiedades fundamentales.
   1) Semigrupos y monoides: Mapeos, orden parcial en semigrupos.
   2) Ideales: Izquierdos, derechos y principales.
   3) Semigrupos: Semigrupos simples y semigrup-0, teorema de Jordan-Holder.
b. Aplicaciones de semigrupos.
   1) Semigrupos de relaciones binarias.
   2) Semigrupos libres y de transformación: Lenguajes y teoría de máquinas.
   3) Semigrupo de substituciones.

c. Teoría de números y computabilidad.
a) Números naturales y enteros.
   1) Postulados de Peano y principio de inducción matemática.
   2) Orden total y buen orden.
   3) Sistema algebraico de los números naturales y enteros.
   4) Algoritmo de la división. Representación de los números enteros.
b) Números, proporción y geometría.
   1) Números de Fibonacci.
   2) Números de Lucas.
   3) Representaciones geométricas.

c) Computabilidad
   1) Recursividad.
   2) Cardinalidad del continuo y conjunto de Cantor.
   3) Palabras infinitas y computación digital infinita.
   4) Curvas de Hilbert y Peano.
   5) Generación de números aleatorios: variedades de números aleatorios.

d. Grupos y sus aplicaciones
   a. Axiomas de grupos.
   b. Generadores y gráficas de grupos.
   c. Grupos de permutación.
   d. Grupos de simetría.
   e. Aplicaciones: geometría, cristalografía y códigos.

Bibliografía

1. Semigrupos y aplicaciones a la computación. Sergio V. Chapa Vergara. Depto. de Ingeniería Eléctrica. Sección de Computación. Notas de clase 2000.
2. Introducción a teoría de grupos. Sergio V. Chapa Vergara. Depto. de Ingeniería Eléctrica. Sección de Computación. Notas de clase 2003.
3. Teoría de números y geometría. Sergio V. Chapa Vergara. Depto. de Ingeniería Eléctrica. Secc. de Computación. Notas de clase 2004.
4. Libro de texto: Kenneth H. Rosen, *Discrete Mathematics and Its Applications*, McGraw-Hill Education (ISE Editions); 6a edition 2007, ISBN: 007288008.

Libros complementarios

1. Ralph P. Grimaldi, *Discrete and Combinatorial Mathematics: An Applied Introduction*, Pearson Addison Wesley, 5a edition 2003 ISBN: 0201726343.
2. Susanna S. Epp, *Discrete Mathematics with Applications*, Thomson Brooks/Cole, 3a edition 2004, ISBN: 0534359450.
3. Judith L. Gersting, *Mathematical structures for Computer Science: A modern approach to Discrete Mathematics*, W. H. Freeman and Company, 6a edition 2006, ISBN: 071676864X.
4. Edgar G. Goodaire y Michael M. Parmenter, *Discrete Mathematics with Graph Theory*, Pearson Prentice Hall, 3a edition 2006, ISBN: 0131679953.
5. Rubin H. Landau, *A first course in Scientific Computing: Symbolic, graphic, and numeric modeling using Maple, Java, and Fortran 90*, Princeton Universtiy Press, 1a edition 2005, ISBN: 0691121834.
6. Edward A. Bender, *Mathematics for algorithm and systems analysis*, Dover Publications, 1a edición 2005, ISBN: 0486442500.

Minería de Datos

Objetivo

Conocer de manera general las técnicas y enfoques del proceso general de Minería de Datos. Se conocen los fundamentos y conceptos necesarios de cada una de las etapas del proceso. Se explora el uso de fuentes de datos para análisis y toma de decisiones resultantes de tareas de clasificación, predicción o agrupamiento.

Contenido

1. Introducción a minería de datos
   a. Objetivos de la minería de datos
   b. Problemas aptos para minería de datos
   c. Aplicaciones comerciales
   d. Aplicaciones no-comerciales
   e. Técnicas generales para análisis de los datos (predicción, clasificación, clustering).

2. Selección de Fuentes de Datos y Calidad de Datos
   a. Datos esperados para minería de datos
   b. Calidad de los datos
   c. Fuentes posibles de datos.

3. Preprocesamiento y Preparación de Datos
   a. Operaciones sobre los datos
   b. Problemas en el manejo de datos reales
   c. Selección de variables
   d. Muestreo, selección de registros
   e. Análisis de correlación
   f. Creación de nuevas variables, agregación de variables

4. Técnicas de Análisis
   a. Principales técnicas para el análisis de datos.
   b. Aplicación de las técnicas de análisis de datos
   c. Técnicas para la identificación de características, tendencias y relaciones en los datos
   d. Visualización
   e. Técnicas estadísticas (correlación, análisis factorial)

5. Creación de modelos de datos: clasificación y predicción
   a. Principales técnicas para clasificación y predicción de datos
   b. Aplicación de las técnicas de clasificación y predicción de datos
   c. Inducción de reglas: C4.5.
   d. Redes neuronales.
   e. Técnicas estadísticas: regresión.

6. Creación de modelos de datos: clustering
   a. Conceptos fundamentales de clustering
   b. Principales técnicas para el clustering
   c. Aplicación de técnicas de clustering

d. Redes neuronales: Kohonen SOM
e. Técnicas estadísticas: K-means
f. Clustering difuso: Fuzzy c-Means
g. Evaluación de modelos
a. Conceptos fundamentales
b. Técnicas de evaluación de modelos
c. Aplicación de las técnicas de evaluación de modelos

Bibliografía

1. Data Mining: Concepts and Techniques, 2nd ed. Jiawei Han and Micheline Kamber Morgan Kaufmann, 2006
2. Data Mining: Practical Machine Learning Tools and Techniques I.H. Witten and E. Frank Morgan Kaufmann, 1999
3. Machine Learning Tom M. Mitchell Prentice Hall, 2003
4. Inteligencia Artificial Russell y Norvig Prentice Hall, 2009
5. Pang-Ning Tan, Michael Steinbach and Vipin Kumar. Introduction to Data Mining. Addison- Wesley. 2006. ISBN: 0321321367.
6. Anil K. Jain, Richard C. Dubes. Algorithms for Clustering Data. Prentice Hall. 1988. ISBN: 013022278X.
7. T. Hastie, R. Tibshirani and J. Friedman. Elements of Statistical Learning: Data Mining, Inference and Prediction. Springer-Verlag. 2001. ISBN: 0387952845.

Optimización en ingeniería

Objetivo

Presentar al estudiante un repaso histórico, teórico y práctico de los diversos métodos de optimización global, enfatizando sus ventajas y desventajas. Así mismo, generar habilidades para decidir y modificar técnicas según las demandas de la aplicación específica. En este curso se estudian diversos métodos de programación matemática para resolver problemas de optimización lineal y no lineal (principalmente sin restricciones). El curso enfatizará aspectos algorítmicos y de implementación sobre los aspectos teóricos, por lo que es necesario tener al menos conocimientos básicos de programación. También se requieren conocimientos de cálculo, trigonometría, geometría y álgebra.

Contenido

1. Introducción a la optimización.
2. Técnicas clásicas.
3. Programación lineal y el método simplex.
4. Programación no lineal: métodos unidimensionales.
5. Programación no lineal: métodos multidimensionales.
6. Programación no lineal: métodos de optimización con restricciones.
7. Repaso de métodos modernos de optimización.

Bibliografía

1. Singiresu S. Rao. Engineering Optimization: Theory and Practice, 5th Edition, John Wiley & Sons, Inc., November 2019, ISBN 978-1119454717.
2. Kalyanmoy Deb. Optimization for Engineering Design: Algorithms and Examples. Ed. Prentice-Hall of India, 1995.
3. David M. Himmelblau. Applied Nonlinear Programming. Ed. McGraw-Hill, 1972.
4. G.V. Reklaitis, A. Ravindran y K.M. Ragsdell. Engineering Optimization: Methods and Applications. Ed. John Wiley & Sons, Inc., 1983.
5. Jorge Nocedal y Stephen J. Wright. Numerical Optimization. Ed. Springer, 1999.
6. Garrett N. Vanderplats. Numerical Optimization Techniques for Engineering Design with Applications. Ed. McGraw-Hill, 1984.
7. J. Frédéric Bonnans, J. Charles Gilbert, Claude Lemaréchal y Claudia A. Sagastizábal. Numerical Optimization: Theoretical and Practical Aspects. Ed. Springer, 2003.
8. P. Venkataraman. Applied Optimization with MATLAB Programming. Ed. John Wiley & Sons, 2002.
9. R. Russell Rhinehart, Engineering Optimization: Applications, Methods and Analysis, Wiley-ASME Press, 2018.
10. Andreas Öchsner and Resam Makvandi, Numerical Engineering Optimization: Application of the Computer Algebra System Maxima, Springer, 2020, ISBN 978-3030433871.

Métodos numéricos

Objetivo

En este curso se proponen y discuten diversos métodos numéricos para resolver varios problemas numéricos clásicos tales como: sistemas de ecuaciones lineales, interpolación y aproximación, encontrar ceros de una función de una variable, integración numérica y cálculo de los eigenvalores de una matriz.

Contenido

a) Análisis de errores
b) Solución de sistemas de ecuaciones lineales
c) Interpolación y Aproximación
d) Ceros de una función
e) Integración numérica
f) Eigenvalores

Bibliografía

1. R. W. Hamming. Numerical Methods for Scientists and Engineers. 2nd edition, Dover Books on Mathematics, 1987, ISBN: 978-0486652412.
2. Steven C. Chapra, Applied Numerical Methods with MATLAB for Engineers and Scientists, 2017, McGraw-Hill, ISBN: 978-0073397962.
3. Qingkai Kong, Timmy Slauw and Alexandre Bayen, Python Programming and Numerical Methods: A Guide for Engineers and Scientists, Academic Press, 2020, ISBN: 978-0128195499.

Optimización numérica

Objetivo

Este curso es la continuación del de “Optimización en Ingeniería”. En este caso, se cubren técnicas avanzadas para manejo de restricciones en problemas escalares. Posteriormente, se discuten los métodos de continuaciones que aparecen en muchos problemas de optimización En la parte final del curso, se proporciona una introducción a la optimización multi-objetivo.

Contenido

a) Programación Cuadrática
b) Métodos de la región de confianza
c) Programación cuadrática secuencial
d) Métodos de Continuación
e) Optimización Multi-objetivo

Bibliografía

1. Jorge Nocedal and Stephen Wright, Numerical Optimization, 2nd edition, Springer, 2006, ISBN: 978-0387303031.
2. Kurt Georg and Eugene L. Allgower, Numerical Continuation Methods: An Introduction, Springer, 2011, ISBN: 978-3642647642.
3. Kaisa Miettinen, Nonlinear Multiobjective Optimization, Springer, 1998, ISBN: 978-1461375449.
4. Andrzej Ruszczynski, Numerical Optimization, Ge Gruyter, 2011.
5. Adil M. Bagirov, Manlio Gaudioso, Napsu Karmitsa, Marko M. Mäkela and Sona Taheri, Numerical Nonsmooth Optimization: State of the Art Algorithms, Springer, 2020, ISBN: 978-3030349127.
6. Fran S. Lobato and Steffen Valder Jr., Multi-Objective Optimization Problems. Concepts and Self-Adaptive Parameters with Mathematical and Engineering Applications, Springer, 2017, ISBN: 978-3319585642.

Programación Concurrente

Objetivo

1. Estudio de los mecanismos para compartir y controlar recursos.
2. Estudio de los mecanismos basados en paso de mensajes.
3. Estudio de lenguajes académicos de programación concurrente: Pascal – S, SR.
4. los conceptos fundamentales en el diseño e implementación de aplicaciones multitarea.
5. Uso de bibliotecas para el desarrollo de aplicaciones multi-hilo: Pthreads, Java-Threads

Contenido

1. Conceptos básicos
    a) Diferencia entre programación secuencial y programación concurrente.
    b) Conceptos de: Proceso, programa multitarea, multiproceso, granularidad, etc.
    c) Ejecución atómica, principio de ejecución concurrente.
    d) Sincronización, no interferencia.
    e) Propiedades: vivacidad, seguridad, exactitud.

2. Exclusión mutua
    a) Introducción al problema de la exclusión mutua.
    b) Propuestas clásicas de solución: una bandera, dos banderas, cesión voluntaria, asignación de turnos.
    c) Especificación de ejecución concurrente: cobegin/coend, fork/join, corutinas.
    d) Concurrencia y sincronización: ejecución intercalada.
    e) Operaciones atómicas: principio de no interferencia
    f) Invariantes globales

3. Mecanismos de control de concurrencia basados en variables compartidas
    a) Semáforos.
    b) Barreras.
    c) Lista de esperas acotadas.
    d) Algoritmos clásicos de la programación concurrente: Productores y consumidores 5 filósofos, barbero dormido, etc.

4. Pascal – S
    a) Importancia histórica de Pascal – S en la evolución de la programación concurrente.
    b) Primer trabajo práctico.

5. Paso de mensajes
    a) Rendez-vous
    b) Llamados a procesos remotos
    c) Llamado a métodos remotos
    d) Comunicación síncrona y asíncrona

6. Lenguaje de programación SR (Sincronizing Resources)
    a) Presentación de SR, como lenguaje académico para el desarrollo de aplicaciones concurrentes
    b) Segundo trabajo práctico

7. Programación multihilo
    a) Introducción a la programación multi-hilo
    b) Diferencia entre hilos y procesos
    c) Gestión de recursos de maquina al programar con hilos
    d) Bibliotecas para la programación multi-hilo
    e) Tercer trabajo práctico

Bibliografía

1. Ben-Ari, M., Principles of Concurrent and Distributed Programming, Prentice Hall, 1990.
2. Lewis, B,. Berg, D., Pthread Primer – A guide to Multithreaded Programming, SunSoft Press 1996.
3. Hoare, C., Communicating Sequential Process, June 2004
4. Goetz, B., Peierls, T., Bloch, J., Bowbeer, J., Holmes, D., Lea, D., Java Concurrency in Practice, Addison Wesley Professional, 2006
5. Kurki-Suonio, Reino (2005). A Practical Theory of Reactive Systems. Springer. ISBN 978-3-540-23342-8.
6. Distefano, S., & Bruneo, D. (2015). Quantitative assessments of distributed systems: Methodologies and techniques (1st ed.). Somerset: John Wiley & Sons Inc.ISBN 9781119131144
7. Tanenbaum, Andrew S.; Van Steen, Maarten (2002). Distributed Systems: Principles and Paradigms. Prentice Hall. ISBN 978-0-13-088893-8.
8. Wolter, K. (2012;2014;). Resilience assessment and evaluation of computing systems (1. Aufl.;1; ed.). London;Berlin;: Springer. ISBN 9783642290329

Procesamiento de Lenguaje Natural

Objetivos:

- Construir modelos de lenguaje natural para su análisis y generación
- Explorar como es el proceso de comunicación empleando el lenguaje natural
- Diseñar y construir herramientas empleadas en el procesamiento automático del lenguaje, por ejemplo: traducción automática, interfaz amigable, gestión de base de datos. Comprensión de voz y texto, sistema de ayuda, corrección de estilo, sistema de comprensión, resolución de problemas, Pre-requisitos,Programar en C.

Descripción

El Proceso de Lenguaje Natural (PLN) trata productos ubicuos: lenguaje empleado en correos electrónicos, páginas web, descripción de productos, periódicos, redes sociales, y artículos científicos.

Los éxitos obtenidos en PLN se han convertido en parte de nuestra experiencia cotidiana, e. g., silabizar, la corrección de la gramática y el estilo en procesadores de texto, la traducción automática, resolución de problemas. De igual manera, se hace uso implícito de detección de "spam" en correos electrónicos, la respuesta automática de encuestas, la determinación de la opinión pública de productos o servicios.

Este curso trata los algoritmos básicos, así como los modelos para procesamiento del lenguaje y como emplear estos conocimientos para resolver problemas prácticos para cualquier aplicación relacionada con datos de lenguaje o comunicación.

Contenido

1. Introducción.
    a. Nivel de abstracción del lenguaje.
    b. Modelos de comprensión del lenguaje.
    c. Algunos ejemplos de aplicación.
2. Proceso del modelo morfológico.
    a. Modelos del lenguaje y n-grama.
    b. Fonemas.
    c. Técnicas de suavizado.
    d. Semántica y ambigüedad de las palabras.
    e. Probabilidad condicionada y teorema de Bayes.
    f. Sistemas de aplicación.
3. Categoría Gramatical.
    a. Parte de la Oración (POS).
    b. Semántica del verbo.
    c. Construcción Nominal, preposicional, pronombre, complementos.
    d. Unidad lógica de la frase.
    e. Diferentes clase Gramaticales.
    f. Técnicas de análisis léxico.
4. Representación del conocimiento.
    a. Simbólica.
    b. Redes Semánticas.

    c. Objetos estructurados.

5. Análisis sintáctico.
1. 5.1 Redes de Transición Aumentada.
2. 5.2 Análisis Estadístico.

6. Análisis semántico.
    a. Dependencia Conceptual.
    b. Descriptores funcionales.
    c. Sentido de las palabras y relaciones entre éstas.
    d. Similaridad entre léxicos.

7. Análisis de Sentimientos.
    a. Análisis de la negación.
    b. Modelos de sistemas de aplicación.

8. Interpretación de preguntas y generación de respuestas.

Bibliografía

1. Akshay Kulkarni, Adarsha Shivananda. Natural Language Processing Recipes: Unlocking Text Data with
2. Machine Learning and Deep Learning using Python. ISBN-13 (pbk): 978-1-4842-4266-7 ISBN-13 (electronic): 978-1-4842-4267-4 (2019). Apress (2019). https://doi.org/10.1007/978-1-4842-4267-4
3. Daniel Jurafsky and James H. Martin. Speech and Language Processing: An Introduction to Natural Language Proceeding, Computational Linguistics, and Speech Recognition (First 2000, Third edition 2018).
4. Alexander Gelbukh, Hiram Calvo. Automatic Syntactic Analysis Based on Selectional Preferences. Studies in Computational Intelligence Vol. 765. ISSN 1860-949X ISSN 1860-9503 (electronic) Studies in Computational
5. Intelligence ISBN 978-3-319-74053-9 ISBN 978-3-319-74054-6 (eBook) https://doi.org/10.1007/978-3-319-74054-6 Library of Congress Control Number: 2018930105. Springer International Publishing AG 2018
6. Li D. Y. Liu (Editors). Deep Learning in Natural Language Processing. ISBN 978-981-10-5208-8 ISBN 978-981-10-5209-5 (eBook) https://doi.org/10.1007/978-981-10-5209-5. Library of Congress Control Number: 2018934459 © Springer Nature Singapore Pte Ltd. 2018.
7. Pramod Singh, Machine Learning with PySpark: NLP and Recommender Systems. ISBN-13 (pbk): 978-1-4842- 4130-1 ISBN-13 (electronic): 978-1-4842-4131-8. Apress. https://doi.org/10.1007/978-1-4842-4131-8 Library of Congress Control Number: 2018966519 Copyright © 2019.
8. Slav Petrov. Coarse-to-Fine Natural Language Processing: Theory and Applications of NLP. Spinger-Verlag. ISBN 978-3-642-22742-4 e-ISBN 978-3-642-22743-1. DOI 10.1007/978-3-642-22743-1. (2012)
9. Steven Bird, Ewan Klein, and Edward Loper. Natural Language Processing with Python. Publisher O'Reilly Media Inc. ISBN: 978-0-596-51649-9 (First 2009. Four edition 2019).
10. Vladimir A. Fomichov. Semantics-Oriented Natural Language Processing. Mathematical Models and
11. Algorithms. IFSR International Series on System Science and Engineering. Volume 27 Springer Verlag. ISBN 978-0-387-72924-4 e-ISBN 978-0-387-72926-8. DOI 10.1007/978-0-387-72926-8. (2010)

Programación Orientada a Objetos

Objetivo

El alumno aprenderá los fundamentos y herramientas de desarrollo de la programación orientada a objetos.

Contenido

1. Fundamentos de Programación Orientada a Objetos
    a. Abstracción.
        a. Objetos como interfaces.
        b. Interfaces e implementación.
        c. Reutilización de código.
        d. Reutilización de interfaces (herencia)
        e. Polimorfismo
    b. Tecnologías de desarrollo.
        a. 1.2.1 Compiladores, cargadores y ligadores (ligado dinámico y estático).
        b. 1.2.2 Sistema en tiempo de ejecución (constructores – destructores, colector de
        c. basura)
        d. 1.2.3. Memoria dinámica (manejo de pila, pool y liberación de memoria).
        e. 1.2.3 Manejo de errores y notificaciones.
2. C++.
    a. Espacio de nombres.
    b. Manejo de iostreams.
    c. Tipos de datos abstractos.
    d. Clases.
    e. Inicialización y destructor.
    f. Sobrecarga de funciones y argumentos por omisión.
    g. Elementos estáticos.
    h. Apuntadores en C++.
    i. Sobrecarga de métodos.
    j. Creación dinámica de objetos
    k. Herencia y composición.
    l. Polimorfismo y funciones virtuales.

3. Java.
    a. Tipos de datos.
    b. Inicialización y borrado.
    c. Control de acceso.
    d. Reutilización de clases.
    e. Polimorfismo.
    f. Interfaces.
    g. Clases internas.
    h. Manejo de errores.
    i. Objetos String y Array.
    j. Manejo de entrada y salida

4. Estructuras de datos.
    1. Arreglos y vectores.
    2. Matrices.
    3. Listas, pilas y colas.
    4. Diccionario de datos y tablas hash.
    5. Árboles (binarios, AVL, Btree y Btree+)

5. Python
    a. Introducción
    b. 5.2. Patrones.
    c. 5.3. Frameworks.
    d. 5.4. Decorador: tipos selección dinámica.
    e. 5.5. Iteradores.
    f. 5.6. Fábricas: encapsulamiento de la creación de objetos.
    g. 5.7. Objetos funcionales.
    h. 5.8. Patrones para cambiar interface (Adapter y Facade).

Bibliografía

1.- Castagna, Giuseppe. Object-Oriented Programming A Unified Foundation. Springer Science & Business Media, 2012.
2.- Lutz, Mark. Learning Python: Powerful Object-Oriented Programming. " O'Reilly Media, Inc.", 2013.
3.-Blaschek, Günther. Object-oriented Programming: With Prototypes. Springer Science & Business Media, 2012.
4.-Drozdek, Adam. Data Structures and algorithms in C++. Cengage Learning, 2012.
5.-Zeigler, Bernard P. Objects and systems: principled design with implementations in C++ and Java. Springer Science & Business Media, 2012.
6.-Josuttis, Nicolai M. The C++ standard library: a tutorial and reference. Addison-Wesley, 2012.
7.-Goodrich, Michael T., Roberto Tamassia, and Michael H. Goldwasser. Data structures and algorithms in Java. John Wiley & Sons, 2014.
8.- Lutz, Mark. Learning Python: Powerful Object-Oriented Programming. " O'Reilly Media, Inc.", 2013.
9.-Lafore, Robert. Data structures and algorithms in Java. Sams publishing, 2017.

Redes de Computadoras

Objetivo

En este curso el alumno conocerá las tecnologías involucradas en las redes de computadoras analizadas a través de la arquitectura por capas siguiendo un enfoque descendente. Presentar los principios básicos de la arquitectura TCP/IP y su implementación en Internet. Se revisarán distintas alternativas de interconexión de redes, la función y problemática de cada una de las capas del modelo ISO/OSI. Se dará especial énfasis a las capas de aplicación, transporte y red del modelo de referencia de Internet.

El alumno conocerá los protocolos básicos dentro de cada capa. Complementara los estudios teóricos con implementaciones básicas de algunos de los algoritmos y protocolos analizados. Analizara algunos aspectos generales de gestión de redes y de nuevas tecnologías de redes inalámbricas.

Contenido

1. Introducción
    a) Elementos de Internet (hosts, routers, conexiones, proveedores, etc.)
    b) Tipos de protocolos (orientación a conexión y sin conexión)
    c) Tipos de redes (difusión, conmutación)
    d) Acceso a Internet. Medios de transmisión.
    e) Rutas y retardos en Internet.
    f) Arquitectura. Modelo de capas. TCP/IP

2. Capa de aplicación
    a) Protocolos básicos
    b) El world wide web: HTTP
    c) Transferencia de archivos: FTP
    d) Correo electrónico y noticias: SMTP, POP3, IMAP, NNTP
    e) Protocolos especializados
    f) Protocolo de configuración dinámica: DHCP
    g) Servicio de directorio: DNS
    h) Protocolo de administración de redes: SNMP
    i) Seguridad: SSL, HTTPS
    j) Programación de Sockets (TCP e UDP)

3. Capa de transporte
    a) Características generales y clasificación
    b) Servicio sin conexión. UDP
    c) Fundamentos de la transferencia fiable
    d) Control de flujo: retransmisión adaptativa, ventana deslizante
    e) Servicio orientado a conexión. TCP
    f) Control de la congestión

4. Capa de red
    a) Técnicas de conmutación: datagramas, circuitos virtuales
    b) Algoritmos de estado de enlaces y vector de distancias
    c) Estrategias de encaminamiento. RIP, OSPF, BGP
    d) El protocolo Internet (IP e IPv6)

e) El protocolo de mensajes de control de Internet (ICMP)
f) Estructura de un router

5. Capa de enlace
   a) Función de la capa de enlace.
   b) Protocolos de acceso al medio. Ethernet.
   c) Hubs, bridges y switches.
   d) El protocolo punto a punto (PPP)
   e) Modo de transferencia asíncrona (ATM)

6. El nivel físico
   a) La transmisión de datos
   b) Señales para la transmisión de datos
   c) Medios de transmisión
   d) Codificación y modulación
   e) Tecnologías de módem

7. Presentaciones adicionales:
   a) Gestión de Redes
   b) Redes Inalámbricas.

Bibliografía

1. Douglas E. Comer and David L. Stevens. 2013. Internetworking with TCP/IP: Principles, Protocols, and Architecture (6rd. ed.). Prentice Hall PTR, USA.
2. S. Tanenbaum, Computer Networks. Prentice Hall, 5th Ed., 2010.
3. W. Stallings, Comunicaciones y Redes de Computadores. Prentice Hall, 7th Ed., 2008.
4. M. Donahoo y K. Calvert, ”TCP/IP Sockets in C: Practical Guide for Programmers (The Practical Guides Series)”,2nd edition Morgan Kaufmann. 2009. ISBN: 1558608265.
5. Computer Networking: A Top-Down Approach, 5th Edition. James F. Kurose, Amhersteith W. Ross. ISBN-10: 0136079679, ISBN-13: 9780136079675. Ed. Addison-Wesley 2010
6. James F. Kurose and Keith W. Ross. 2012. Computer Networking: A Top-Down Approach (6th Edition) (6th. ed.). Pearson.
7. W. Stevens, TCP/IP Illustrated, Vol. 1: The Protocols, Addison-Wesley, 2nd Edition 2011
8. Página del Internet Engineering Task Force http://www.ietf.org

Seguridad en sistemas de información

Objetivo

Presentar políticas generales de autenticación, controles de acceso, protección a sistemas de archivos, respaldos y algunas otras nociones.

Descripción

Es un curso compartido por varios profesores y tiene componentes teóricos y prácticos, propios de ingeniería computacional y de protocolos ad-hoc en cuanto a aplicaciones.

Contenido

1. Seguridad en comunicaciones y redes de computadoras
    a. Introducción a TCP/IP
    b. Conceptos de seguridad en redes
    c. Configuración de red en el sistema GNU/Linux
    d. Configuración de una puerta (gateway)
    e. Uso de IPTables
    f. Configuración de cortafuegos
    g. Zonas militarizadas y redireccionamiento de servicios
    h. Configuración y compilación del núcleo de GNU/Linux
    i. Cómo crear un cortafuegos sin un disco duro. Arranque desde CDROM.
    j. LDAP
    k. Seguridad en redes inalámbricas:
    l. autenticación de usuarios y un cortafuego dinámico: NoCAT y WifiDog
    m. Redes virtuales

2. Aplicaciones en áreas profesionales
    a. Servicios de Seguridad en Sistemas de Información: Introducción.
    b. Servicios de Seguridad en Sistemas de Información: Aplicaciones.
    c. Casos de Estudio: Notaría Digital, Elecciones electrónicas, dinero digital
    d. Protocolo IKE de IPSec: Sigma.
    e. Certificados e Infraestructura de clave Pública (PKI)
    f. Caso de estudio 1: Certificados con información biométrica.
    g. Caso de estudio 2: Facturas electrónicas del SAT
    h. Protocolo SSL/TLS.
    a. Open SSL
    i. PGP
    j. Seguridad en el cómputo nube
    k. Seguridad en Ambientes Computacionales Altamente Restringidos.
    l. Seguridad en Redes Inalámbricas de Sensores.

3. Redes de computadoras
    a. Análisis de Tráfico en Redes Locales
    b. SNMP
    c. Control de Acceso
    d. Autentificación mediante Kerberos y otros protocolos

e. Instalación y configuración de un servidor de correo electrónico con políticas de seguridad
f. Mecanismos Anti-SPAM para el Correo Electrónico
g. Servicios de seguridad en SMS
h. Sistemas de Detección de Intrusos

Bibliografía

a. Alexander Clemm. Network Management Fundamentals. Cisco Press. 2006.
b. De la Fraga, L. G. Seguridad en Redes de Computadoras Usando GNU/Linux. Notas del curso impartido el 9 de septiembre 2004 en el 1st International Conference on Electrical and Electronics Engineering. Acapulco, Guerrero. September 8-10, 2004.
c. Brian Hayes, "Cloud Computing", Communications of the ACM July 2008 Vol. 51 No 7.
d. Gerhard Mourani, Securing & Optimizing Linux: The Ultimate Solution gmourani@openna.com, version 2.0, July 2002 http://www.tldp.org
e. Thomas Wadlow and Vlad Gorelik, "Security in the Browser", Communications of the ACM, Vol. 52 No. 5, May 2009.
f. Ryan West, "The Psychology of Security", Communications of the ACM, Vol. 51 No. 4, April 2008.
g. William Stallings. Cryptography and Network Security: Principles and Practice, 5th Edition. Prentice Hall. 2011.
h. Stallings, Cryptography and Network Security. Principles and Practice, Third Edition.W. Prentice-Hall, 2003.
i. Securing & Optimizing Linux: The Ultimate Solution Gerhard Mourani, gmourani@openna.com, version 2.0, July 2002 http://www.tldp.org
j. Sitio de OpenSSL (www.openssl.org)

Referencias adicionales

1. Christian Cachin, Protocols for Secure Cloud Computing, IBM, April 2011
2. Estrategia Nacional de Ciberseguridad, Gobierno de México, 2017
3. Open Web Application Security Project (OWASP), Top 10 - 2017, The TenMost Critical Web Application Security Risks
4. Open Web Application Security Project (OWASP), Software Assurance Maturity Model (SAMM) ver. 2.0 (Local copy)
5. RedHat Developer, Secure Coding. Includes videos for Getting Started with Secure Coding, but the most interesting part is The Fedora Project's Defensive Coding Guide
6. Serie de plantillas para definir políticas de seguridad informática, recomendadas por el Instituto SANS (SysAdmin, Audit, Network and Security).
7. William Stallings and Lawrie Brown. Computer Security: Principles and Practice. Prentice Hall Press, USA, 3rd edition, 2014.
8. Avi Kak, TCP/IP Vulnerabilities and DoS Attacks: IP Spoofing, SYN Flooding, and The Shrew DoS Attack, Lecture Slides, 2020

Sistemas Colaborativos Distribuidos

Objetivo

Dar a conocer al alumno los fundamentos teóricos y prácticos del campo de investigación multidisciplinario denominado "Trabajo Cooperativo Asistido por Computadora" (*CSCW por sus siglas en inglés*), haciendo énfasis en el estudio de los sistemas computacionales (*Groupware* por su denominación en inglés) que soportan grupos de personas comprometidas en un proyecto común y que proveen una interfaz a un entorno compartido. En particular, se analizan las arquitecturas de distribución fundamentales para permitir a personas físicamente distribuidas comunicar, colaborar y coordinar sus actividades como si estuvieran cara a cara. Asimismo, se estudian los principales mecanismos propuestos para administrar la compartición de la información, tanto a nivel de interfaz de grupo como a nivel de núcleo funcional. Este dominio de investigación ha contribuido a la evolución de diversos dominios de aplicación, entre los que se encuentran los sistemas de mensajes, los editores de grupo, los sistemas de soporte para la toma de decisiones en grupo, las salas de reuniones virtuales, las conferencias por computadora, los agentes inteligentes, los sistemas de coordinación (*workflows*) y la enseñanza/aprendizaje colaborativo.

Contenido

1. Introducción al Trabajo Cooperativo Asistido por Computadora (TCAC)
    a. Motivación, historia y objetivos
    b. Perspectivas del TCAC:
        i. Sociológica
        ii. Sistemas distribuidos y bases de datos
        iii. Redes de comunicaciones
        iv. Interacción hombre-máquina
        v. Inteligencia artificial
    c. Diferencias entre aplicaciones mono-usuario, multi-usuario y colaborativas
    d. Modelo conceptual de sistemas colaborativos
        i. Espacio de comunicación
        ii. Espacio de producción
        iii. Espacio de coordinación

2. Topologías de sistemas colaborativos
    a. Interacciones entre colaboradores
        i. Cara a cara
        ii. Síncronas-remotas
        iii. Asíncronas
        iv. Asíncronas-remotas
    b. Ejemplos representativos de aplicaciones

3. Requerimientos de los sistemas colaborativos
    a. Sincronía de acciones
    b. Grano de la información compartida y del tiempo
    c. Administración de la información compartida
        i. Arquitecturas de distribución
        ii. Mecanismos de actualización

iii. Políticas de control de acceso
iv. Estrategias de control de concurrencia
v. Persistencia

d. Infrastructuras (frameworks) vs cajas de herramientas (toolkits)

4. Problemas en el diseño de sistemas colaborativos
   a. Interfaces de grupo
      i. Vistas WYSIWIS estrictas y relajadas
      ii. Foco de atención y distracción
      iii. Adaptabilidad a la dinámica del grupo
      iv. Administración del espacio de despliegue
      v. Ejemplos de elementos de interfaz de grupo (widgets)
   b. Procesos de grupo
      i. Acceso secuencial vs. paralelo
      ii. Protocolos tecnológicos y sociales
      iii. Operaciones síncronas vs. asíncronas
   c. Mecanismos de control de concurrencia
      i. Candados explícitos
      ii. Transacciones
      iii. Copias maestras/esclavas
      iv. Candados flexibles
      v. Toma de turno
      vi. Detección de dependencias
      vii. Ejecución reversible
      viii. Transformación de operaciones
   d. Control de acceso
   e. Notificación de eventos

5. Arquitecturas de software para sistemas colaborativos
   a. Presentación-Abstracción-Control (PAC*)
   b. Abstracción-Liga-Vista (ALV)
   c. Modelo-Vista-Controlador (MVC)
6. Arquitecturas de distribución
   a. Centralizada
   b. Totalmente replicada
   c. Hybrida
   d. Semi-replicada
   e. Asíncrona
   f. Arquitecturas adaptables

Bibligorafia

1. Ziang Xiao, Michelle X. Zhou, Q. Vera Liao, Gloria Mark, Chang Yan Chi, Wenxi Chen, and Huahai Yang, “Tell Me About Yourself: Using an AI-Powered Chatbot to Conduct Conversational Surveys with Open-ended Questions”, ACM Transactions on Computer Human Interaction, 27(3): 15:1-15:37 (2020).
2. Dakuo Wang, Elizabeth F. Churchill, Pattie Maes, Xiangmin Fan, Ben Shneiderman, Yuanchun Shi, Qianying Wang, From Human-Human Collaboration to Human-AI Collaboration: Designing AI Systems That Can Work Together with People. CHI Extended Abstracts 2020: 1-6.

3. Jean Vanderdonckt, Sara Bouzit, Gaëlle Calvary, Denis Chêne, “Exploring a Design Space of Graphical Adaptive Menus: Normal vs. Small Screens” ACM Transactions on Interactive Intelligent Systems 10(1): 2:1-2:40 (2020).
4. Jonathan Grudin, Richard Jacques, “Chatbots, Humbots, and the Quest for Artificial General Intelligence”, CHI 2019: 209.
5. Nigel Davies, Marc Langheinrich, Pattie Maes, Jun Rekimoto, “Augmenting Humans”. IEEE Pervasive Computing, 17(2): 9-10 (2018).
6. Paul Dourish, “User experience as legitimacy trap”, Interactions 26(6): 46-49 (2019).
7. Benjamin J. Lafreniere, Carl Gutwin, and Andy Cockburn, “Investigating the Post-Training Persistence of Expert Interaction Techniques”, ACM Transactions on Computer-Human Interactions, 24(4): 29:1-29:46 (2017).
8. Jonathan Grudin, “From tool to partner: The Evolution of Human-Computer Interaction”, Synthesis Lectures on Human-Centered Interaction, 10(1): 1-183, Morgan & Claypool Publishers (2017).
9. Saul Greenberg and Carl Gutwin, “Implications of We-Awareness to the Design of Distributed Groupware Tools”, Computer Supported Cooperative Work, 25(4-5): 279- 293 (2016).
10. Marina Cidotã, Stephan G. Lukosch, Dragos Datcu, Heide K. Lukosch, “Workspace Awareness in Collaborative AR using HMDs: A User Study Comparing Audio and Visual Notifications”, AH 2016: 3:1-3:8.
11. Dhaval Vyas, Alan J. Dix, and Gerrit C. van der Veer, “Reflections and Encounters: Exploring Awareness in an Academic Environment”, Computer Supported Cooperative Work, 24(4): 277-317 (2015).
12. Stephan G. Lukosch, Mark Billinghurst, Leila Alem, Kiyoshi Kiyokawa, “Collaboration in Augmented Reality”, Computer Supported Cooperative Work. 24(6): 515-525 (2015).
13. Andy Cockburn, Carl Gutwin, Joey Scarr, and Sylvain Malacria, “Supporting Novice to Expert Transitions in User Interfaces” ACM Comp. Surveys 47(2): 31:1-31:36 (2014).

Sistemas Distribuidos

Objetivo

Dotar al alumno de conocimientos para que pueda comprender y aplicar los sistemas distribuidos, tanto en el área de base de datos, como en el área de aplicaciones de red, como manejo de protocolos, sistemas operativos, bajo diferentes tipos de enlaces, diferentes arquitecturas de cómputo distribuido(cliente/servidor). Se analizarán las principales tecnologías de programación para sistemas distribuidos utilizando Middlewares como RPCs, RMIs, Corba y Servicios Web.

Contenido

1. Introducción a los Sistemas Distribuidos
    a. Caracterización de los sistemas distribuidos
    b. Ejemplos de sistemas distribuidos
    c. Recursos compartidos
    d. La Web

2. Modelos de sistema
    1. Modelos arquitectónicos
    2. Modelos fundamentales

3. Comunicación en sistemas distribuidos
    1. Mecanismos básicos de comunicación (IPC)
    2. Comunicación cliente/servidor
    3. Sockets y RPCs
    4. Presentación externa de datos y empaquetado (xdr, cdr, xml)
    5. Comunicación en grupo
    6. Middlewares
    7. Manejo de procesos e hilos

4. Sistemas de archivos distribuidos
    a) Conceptos básicos y estructura
    b) Servicios de directorio y archivos
    c) Replicación
    d) Manejo de transacciones
    e) Caso de estudio: NFS, AFS

5. Sincronización y coordinación
    a. Mecanismos de sincronización entre procesos
    b. Algoritmos de sincronización de relojes
    c. Manejo de estado global
    d. Coordinación y acuerdo: exclusión mutua, elección y multidifusión

6. Objetos distribuidos e invocación remota (middlewares)
    a. Introducción
    b. Comunicación entre objetos distribuidos
    c. Llamadas a procedimientos/métodos remotos (RPC/RMI)
    d. Eventos y notificaciones

e. RMIs

7. Aplicaciones Web
    1. Aplicaciones con clientes pesados (desarrollo del lado del cliente: DHTML)
    2. Aplicaciones con clientes ligeros (desarrollo del lado del servidor: JSP)
    3. Arquitectura de aplicaciones Web usando MVC
    4. Servicios Web

8. Seguridad
    1. Introducción
    2. Técnicas de seguridad: firewalls, criptografía, certificados
    3. Algoritmos criptográficos: simétricos y asimétricos
    4. Firmas digitales

Bibliografía

1. Distributed Systems: Concepts and Design. G. Coulouris, J. Dollimore, T. Kindberg, Addison Wesley, 2017, 5th edition. ISBN: 978-9332575226.
2. From P2P to Web Services and Grids: Peers in a Client/Server World, Ian J. Taylor and Andrew Harrison, Springer; 2006, ISBN 978-1852338695
3. Distributed Systems: Principles and Paradigms, Andrew S. Tanenbaum, Maarten van Oteen, Prentice Hall; United States, 2nd edition, 2016, ISBN: 0132392275
4. Concurrent and Distributed Computing in Java, Vijay K. Garg, Wiley-IEEE Press, 2007, ISBN: 047143230X
5. Java in Distributed Systems: Concurrency, Distribution and Persistence, Marko Boger, John Wiley & Sons; 1 edition, 2001, ISBN: 0471498386
6. Distributed Operating Systems. Andrew S. Tanenbaum, Prentice Hall; USA, Second Edition, 2006, ISBN: 978-0132392273.
7. Distributed Operating Systems: Concepts and Practice Doreen L. Galli, Prentice Hall; 1st edition (August 31, 1999) ISBN: 0130798436.
8. Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable Services, Brandon Burns, O'Reilly Media, 2018, ISBN: 978-1491983645

Sistemas de tiempo real

Objetivo

En los sistemas de tiempo real existen 3 componentes principales que los caracterizan, así como la interrelación entre estos. En primer lugar, está el tiempo, el cual es el recurso más valioso a tratar y gestionar en un sistema en tiempo real. Las tareas deben ser asignadas y planificadas a fin de que cumplan unos plazos (*deadlines*). La correcta ejecución del sistema de tiempo real depende tanto de la validez lógica de la respuesta, como del instante de tiempo en que se produce. Como segundo componente se encuentra la confiabilidad, debido a que un fallo en el sistema en tiempo real podría causar serias consecuencias. Estos dos componentes son la principal causa de una mayor dificultad en el diseño de estos sistemas respecto a los sistemas informativos de propósito general. El tercer componente involucra al ambiente en el cual el computador opera, el cual es un componente activo en cualquier sistema de tiempo real. Por ejemplo, en un sistema de avión, no tiene sentido considerar el sistema de cómputo por separado de la aeronave, ya que ambos interactúan entre sí.

Contenido

1. Introducción a los Sistemas de Tiempo Real (* nuevo)
2. Aplicaciones de Sistemas de Tiempo Real.
3. Sistemas Operativos.
4. Sistemas Operativos: Componentes y Ejecutivo de Tiempo Real.
5. Procesos Concurrentes.
6. Sincronización de Procesos.
7. Modelo del Sistema
8. Métodos de Planificación
9. Planificación Cíclica
10. Planificación por Prioridades Fijas
11. Planificación Dinámica de Tareas
12. Planificación de Tareas Aperiódicas
13. Overload Scheduling in Real-Time Systems
14. Interacción entre Tareas
15. Tolerancia a Fallos
16. Introduction to Embedded Systems

Sistemas operativos

Objetivo

Este curso aborda el diseño y la implementación de un sistema operativo y el conocimiento de los tópicos avanzados en sistemas operativos distribuidos.

Contenido

1. Introducción
    - 1.1 Esquema general de computadora
    - 1.2 Componentes
    - 1.3 Estructuras de Sistemas Operativos

2. Procesos
    - 2.1 Concepto de proceso
    - 2.2 Calendarización
    - 2.3 Comunicación entre procesos
    - 2.4 Hilos
    - 2.5 Sincronizacion entre procesos

3. Manejo de memoria
    - 3.1 Memoria principal
    - 3.2 Modelos de manejo de memoria
    - 3.3 Memoria Virtual

4. Estructura de almacenamiento
    - 4.1 Estructura de discos
    - 4.2 Despacho de trabajo en discos
    - 4.3 manejo de disco
    - 4.4 Manejo de espacio de intercambio
    - 4.4 Estructura RAID

5. Sistema de Archivos
    - 5.1 Concepto de Archivo
    - 5.2 Métodos de Acceso
    - 5.3 Directorios y estructura de disco
    - 5.4 Protección
    - 5.5 Interfaz de llamados al sistema y VFS

6. Tópicos avanzados
    - 6.1 Sistemas Distribuidos
    - 6.2 Manejo de Comunicación entre Procesos Memoria
    - 6.3 Almacenamiento (Paralelos & Hadoop)
    - 6.4 Virtualización en Cloud Computing
    - 6.5 Manejo de contenedores

    Servicios (Infraestructura y energía)

Bibliografía

a. Per Brinch Hansen. “Operating Systems Principles”. Prentice Hall, (2001).
b. Silberschatz, Abraham, Peter Baer Galvin, and Greg Gagne. Operating system concepts essentials. John Wiley & Sons, Inc., 2014.
c. Books, L. L. C. "Mobile Phone Operating Systems: Symbian Os, Android, Mobile Operating System, Webos, S60, Cyanogenmod, Symbian Platform, Blackberry OS, 2010.
d. Lashkari, Arash Habibi, and Mohammadreza Moradhaseli. “Mobile operating systems and programming: mobile communications”. VDM publishing, (2011).
e. Sinha, Pradeep K. Distributed operating systems: concepts and design. PHI Learning Pvt. Ltd., 1998.
f. Chow, Randy, and Yuen-Chien Chow. Distributed operating systems and algorithms, Addison-Wesley Longman Publishing Co., Inc., 1997.
g. Manvi, Sunilkumar S., and Gopal Krishna Shyam. "Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey." Journal of Network and Computer Applications 41 (2014): 424-440.
h. Chandio, Aftab Ahmed, et al. "To investigate classical scheduling schemes with power management in IaaS cloud environment for HPC workloads." Research and Development (SCOReD), 2017 IEEE 15th Student Conference on. IEEE, 2017.
i. Takano, Ryousei, et al. "AIST Super Green Cloud: lessons learned from the operation and the performance evaluation of HPC cloud." International Symposium on Grids and Clouds. Vol. 15. No. 20. 2015.
j. Egwutuoha, Ifeanyi P., and Shiping Chen. "Cost of Using Cloud Computing: HaaS vs. IaaS." Handbook of Research on End-to-End Cloud Computing Architecture Design (2016): 455.
k. Kim, Jongyeop, et al. "Performance evaluation and tuning for MapReduce computing in Hadoop distributed file system." Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on. IEEE, 2015.
l. Shahabinejad, Mostafa, Majid Khabbazian, and Masoud Ardakani. "An efficient binary locally repairable code for hadoop distributed file system." IEEE Communications Letters 18.8 (2014): 1287-1290.
m. Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau, Operating Systems: Three Easy Pieces, Createspace Independent Publishing Platform, 2018, ISBN: 978-1985086593.

Tópicos Selectos de Redes Complejas y Aprendizaje Computacional

Objetivo

En el curso se estudian algoritmos para el análisis y modelado de redes complejas, así como para el aprendizaje automático de estrategias de juegos complejos. Y las relaciones entre ambas temáticas.

Para el modelado formal de redes complejas un objetivo fundamental es analizar las interacciones entre nodos (individuos), y las estructuras emergentes a partir de la interacción, considerando parámetros como cercanía, condiciones de cohesión y agrupación, diámetros y centralidad. Se estudian los modelos y algoritmos de simulación para redes complejas, clásicas y emergentes. Asimismo, para automatizar juegos de estrategia como el Go (de tablero), se requieren algoritmos de aprendizaje para el reconocimiento de patrones, paralelos y distribuidos, de colaboración y competencia, de correlación entre lo local y lo global. Ambos, redes y juegos complejos son marco de desarrollo y prueba de algoritmos relevantes en la Inteligencia Computacional con aplicaciones en problemas científicos actuales.

Contenido

1. Redes complejas:
    - 1.1. Conceptos básicos y ejemplos:
        - 1.1.1. Agrupamiento, centralidad, cohesión, diámetro.
        - 1.1.2. Redes de muro pequeño.
    - 1.2. Redes libres de escala
    - 1.3. Redes modeladas con una ecuación maestra:
        - 1.3.1. Distribución del grado entre nodos
        - 1.3.2. Comportamiento cíclico

2. Juegos.
    a. Estratégicos:

1. De múltiples jugadores
2. De tablero.
    b. En forma normal
    c. Competitivos, cooperativos
    d. Serios
    e. El concepto del equilibrio en juegos:

(1) El equilibrio del Nash
(2) El equilibrio Kantiano
(3) Teoría del valor de Shapley paa coaliciones
(4) Ejemplo: Football Americano

3. Algoritmos de aprendizaje con redes neuronales de:
    a. Retro-propagación
    b. Asociativas
    c. Estocásticas: de Hopfield y de Ising
    d. Con estadísticas
    e. Con funciones de distribución de probabilidad
    f. Auto- organizadas de Kohonen
    g. Ejemplo: el juego de Go

a. Tácticas y estrategias

b. Razonamiento estratégico

4. Inteligencia computacional en redes y juegos complejos:
    a. Sistemas de multi-procesamiento:
a. Distribución de tareas
b. Balance de carga
    1. Redes sociales:
        i. Estructuras
        ii. Grupos
        iii. Jerarquías

Bibliografía

a. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, Algorithmic Game Theory, Cambrige University Press, 2007.
b. Nathan Sturtenat, Multi-Player Games: Algorithms and approaches, PhD Thesis, University of California, Los Angeles, 2003.
c. R. Myres , Game Theory: Analysis of Conflict. Harvard University Press 1991.
d. Alvarado M., Yee, A. And Fernández, J., Simulation of American Football Gaming. In CCCS 2013, International Conference on Sport Science and Computer Science, Hong Kong, 2013
e. Yee, A. And Alvarado, M. , Pattern Recognition and Monte-Carlo Tree Search for Go Gaming Better Automation. In IBERANIA 2012: LNCS 7637, pp11-20
f. Raúl Rojas, Neural Networks: a Systematic Study. Springer Verlag
g. R. Alvarez-Martínez, G. Cocho, R. F. Rodríguez, G. Martínez-Mekler. Birth and Death Master Equation for the Evolution of Complex Networks. Elsevier, 2013.
h. Vito Latora, Vincenzo Nicosia and Giovanni Russo, Complex Networks: Principles, Methods and Applications, Cambridge University Press, 2017, ISBN: 978-1107103184.

Tópicos Selectos de Computación Científica

Objetivo

Estudiar la teoría y los métodos matemáticos-computacionales para la resolución de algunos problemas de ingeniería, física, química y biología.
Estudiar los modelos y métodos matemáticos computacionales haciendo énfasis en aspectos algebraicos, geométricos y de visualización.

Descripción

Se hace énfasis en ecuaciones lineales de diverso orden, abordando el problema de condiciones iniciales y el problema de valores a la frontera. El problema de Sturm-Liouville es tratado como uno de valores a la frontera regular y como un problema de valores propios. Se estudia el método de Monte Carlo para simulación y solución de algunos problemas científicos y de ingeniera.
Se estudia la teoría de grupos de matrices de rotación con aplicación a algunos problemas físicos, la visualización de geometría del espacio fase en las soluciones de ecuaciones diferenciales, los autómatas celulares y los sistemas dinámicos discretos como ambientes para modelar problemas físicos y de ecosistemas.

Contenido

1. Espacios lineales y grupo de matrices
    a. Geometría de los espacios lineales
    b. Estructura de matrices
    c. Grupo de matrices
2. Herramientas básicas del algebra lineal
    a. Representación y solución de sistemas lineales
    b. Factorización LU
    c. Sistema tridiagonal y pentagonal de ecuaciones
    d. Solución al problema de valores propios
3. Ecuaciones diferenciales ordinarias
    a. El problema de valores iníciales de EDO
    b. El problema de valores a la frontera y de Sturm-Liouville
4. Métodos numéricos para ecuaciones diferenciales
    a. Métodos para el problema de valores iniciales
    b. Computación de la exponencial matricial
    c. Métodos para el problema de valores a la frontera
5. Método Monte Carlo
    a. Generación de números aleatorios
    b. Generación de diversas variables aleatorias continúas
    c. Generación de diversas variables aleatorias discretas
    d. Pruebas Estadísticas
    e. Importancia de muestreo
6. Experimentación y modelos
    a. Modelos basados en ecuaciones diferenciales de segundo orden
    b. Modelos basados en Monte Carlo
7. Grupo de matrices de rotación
    a. Función de una matriz

b. Series de Campbell-Hausdorff
c. Mapeo exponencial de matrices simétricas y antisimétricas
d. Geometría del grupo de rotación en 3 dimensiones

8. Visualización de Variable compleja
    a. Funciones complejas
    b. Transformación de Mobius
    c. Esfera de Riemann
    d. Funciones especiales
9. Visualización y geometría del espacio fase
    a. Matrices unimodulares y diagramas de bifurcación
    b. Partición en el espacio fase R3
    c. Llenado del espacio del campo direccional en R3
    d. Secuencias de rotación en el espacio fase
    e. Geometría simpectica y el espacio fase
10. Sistemas dinámicos y Autómatas celulares
    a. Autómatas celulares lineales
    b. Autómatas celulares reversibles
    c. Computabilidad y autómatas celulares
    d. Comportamiento no trivial y Chate Maneville
    e. Modelos de reacción y difusión
    f. Modelos de coexistencia de dos especies

Bibliografía

1. S.E. Arge, A.M. Bruaset, H.P. Langtangen; "Modern Software Tools for Scientific Computing"; Birkhauser; 1997.
2. Dæhlen, A. Tveito; "Numerical Method and Software Tools in Industrial Mathematics"; Birkhauser; 1997.
3. Harold V. McIntosh; "Linear Cellular Automata"; Universidad Autónoma de Puebla; Puebla, México; May 20, 1987
4. Harold V. McIntosh; "Linear Cellular Automata via de Bruijn Diagrams"; Universidad Autónoma de Puebla; Puebla, México; May 20, 1990.
5. Tristan Needham; "Visual Complex Analysis"; Oxford University Press; 1997.
6. T. Toffoli, N. Margolus; "Cellular Automata Machines"; MIT Press, 1987.
7. Sergio V. Chapa Vergara, Harold V. McIntosh; "Ecuaciones Diferenciales Ordinarias y Teoría de Weyl, Tomo I: Ecuaciones Diferenciales Escalares y Problema de Sturm- Liouville"; Ed. Lagares, México D.F. 2005
8. Sergio V. Chapa Vergara, Harold V. McIntosh, Amílcar Meneses Viveros; "Ecuaciones Diferenciales Ordinarias y Teoria de Weyl, Tomo III: Teoría de Weyl y aplicaciones a la mecánica cuántica"; por publicar.
9. Harold V. McIntosh; "Complex Analysis"; Universidad Autónoma de Puebla; Puebla, México; April 20, 2001.
10. Bernd Thaller; "Visual Quantum Mechanics"; Springer-Verlag, July 2000.
11. Bernd Thaller; "Advanced Visual Quantum Mechanics" Springer-Verlag, 2005.
12. David P. Landau, Kurt Binder; "A Guide to Monte Carlo Simulations in Statistical Physics"; Cambridge University Press; 2000.
13. J.D. Hoffman; "Numerical Methods for Engineers and Scientists"; Marcel Dekker; Second Edition, 2001.
14. S. Wolfram; "A New Kind of Science"; Wolfram Media, 2002.

Tópicos Selectos en Cómputo Móvil

Objetivo

Revisar tópicos más recientes relativos al cómputo móvil.

Contenido:

1. Arquitecturas avanzadas de sistemas móviles.
2. IP-móvil y problemas asociados a la movilidad.
3. Estándares para el desarrollo de aplicaciones de Web móvil.
4. Interacción humano-computadora para dispositivos móviles.
5. Diseño de aplicaciones abiertas y de plataformas cruzadas.
6. Estrategias de manejo de restricciones en dispositivos móviles.

Bibliografía

1. Mobile and Wireless Desing Essentials, Martyn Mallick, John Wiley & Sons 2003.
2. IP Desing forn Mobile Netwoeks Grayson, M. Shatzkamer, Wainner, S. Cisco Press 2009.
3. Designing Mobile Service Systems, Els Van de Kar Alexander Verbraeck, IOS Press, 2007.
4. Steve Love; “Understanding Mobile Human-Computer Interaction”; Elsevier, 2005.
5. Y. Rogers, H. Sharp, J. Preece; “Interaction Design: Beyond Human-Computer Interaction”; Jhon Wiley & Son Ltd. 2011.
6. Hoang T. Dinh, Chonho Lee, Dusit Niyato, and Ping Wang. A survey of mobile cloud computing: architecture, applications, and approaches. Wireless Communications and Mobile Computing, 2011.
7. Dimitrios Tzovaras, editor. Multimodal User Interfaces: From Signals to Interaction. Springer, Berlin, 2008.
8. M. Pilgrim. HTML5: Up and Running. Oreilly Series. O’Reilly Media, 2010

Tópicos Selectos en Inteligencia Artificial: Agentes y Multiagentes

Objetivo

Los sistemas multiagentes surgieron en el campo de la investigación de tecnología de la información en la década de los 90. Un agente es un sistema o componente de software, el cual es capaz de cooperar para resolver problemas específicos. El objetivo del curso es dar una visión introductoria a los agentes autónomos y a los sistemas multiagentes desde el punto de vista teórico como práctico. Se explicarán las diferentes arquitecturas de agente (reactiva, deliberativa e hibrida), así como los mecanismos de interacción, coordinación y cooperación entre sistemas multiagentes. Las aplicaciones son diversas: control de procesos industriales, comercio electrónico, subastas, etc.

Objetivos específicos:

a. Comprender los diferentes enfoques de la Inteligencia Artificial distribuida
b. Estudiar los diferentes modelos de sistemas de agentes
c. Diseñar y construir un prototipo que muestre el razonamiento llevado a cabo para resolver
d. problemas de acuerdo con alguno de los modelos estudiados.

Contenido

1. Introducción
    a) Concepto de agente.
    b) Agentes y objetos.
    c) Agentes y sistemas expertos.
    d) Agentes y sistemas distribuidos.
    e) Campos de aplicación típicas.

2. Agentes Inteligentes
    a) Arquitecturas abstractas para agentes.
    b) Diseño de agentes inteligentes.
    c) Mecanismo de razonamiento.
    d) Agentes como sistemas reactivos
    e) Arquitectura hibrida.

3. Mecanismos de Inferencia
    a) Demostración de teoremas
    b) Programación orientada a agentes
    c) Lógicas para sistemas multiagentes
    d) Lógica modal

4. Sistemas multiagentes
    a) Interacción entre agentes: principios de la cooperación.
    b) Sistemas cooperación vs. No cooperativos.
    c) Heurísticas para la cooperación
    d) Coherencia y coordinación
    e) Negociación y argumentación
    f) Aplicaciones: subasta, comercio electrónico
    g) Criterios de evaluación

5. Comunicación
    a) Lenguajes de comunicación de agentes
    b) Protocolos KQML /KIF
    c) Ejemplos de aplicación

Bibliografía

1. Rafael H. Bordini et al., Multiagent programming: Languages, platforms, and applications, Springer, New York, 2005.
2. Jaques Ferber, Multiagent Systems. An introduction to Distributed Artificial Intelligence, Addison Wesley, NY, 1999.
3. Ronald Fagin, Reasoning about knowledge, Cambridge Mass: MIT Press, 1995.
4. M. Fisher and M. Wooldrige, “On the formal specification and verification of multiagent systems”, International Journal of Cooperative Information Systems, 6(1), pp. ,1997.
5. Ana Mas, Agentes de software y sistemas multiagentes, Pearson-Prentice Hall, UK, 2005.
6. Michael Wooldridge, An Introduction to Multiagent Systems, John Wiley, 2002.
7. FIFA, “Fifa specification version 2.0”, Technical report, FIPA, part (2), Foundation for Intelligent Physical Agtent 1998.
8. M. Wooldrige and N.R. Jennings, “Intelligent Agents: Theory and Practice”, The Knowledge Engineering Review, Vol 10(2), pp. 115-152, 1995.
9. Hongjing Liang and Juaguang Zhang, Cooperative Tracking Control and Regulation for a Class of Multi-Agent Systems, Springer, 2019, ISBN: 978-9811383618.
10. Olivier Boissier, Rafael H. Bordini, Jomi Hubner and Alessandro Ricci, Multi-Agent Oriented Programming: Programming Multi-Agent Systems Using JaCaMo, The MIT Press, 2020, ISBN: 978-0262044578.

Tópicos selectos en Inteligencia Artificial: Introducción a la optimización evolutiva multiobjetivo

Objetivo

En este curso se estudiarán los conceptos básicos de la optimización multiobjetivo, así como el uso de los algoritmos evolutivos en esta área. El material cubierto abarca desde los orígenes de la optimización multiobjetivo (en economía y planeación), hasta los avances más recientes. Además de analizar las técnicas evolutivas multiobjetivo de mayor uso en la actualidad, se estudiarán otras heurísticas que también han sido extendidas para lidiar con problemas multiobjetivo (p.ej., la colonia de hormigas), discutiendo sus ventajas y limitantes principales. Adicionalmente, se revisará el trabajo teórico realizado en esta área y se discutirán algunos de los temas de investigación futura que han permanecido poco explorados durante los últimos años.

Contenido

1. Conceptos Básicos
    a) Atributos, metas, criterios y objetivos
    b) Definición de un problema multiobjetivo
    c) Tipos de problemas multiobjetivo
    d) Vector ideal
    e) Convexidad y concavidad
    f) Optimo de Pareto
    g) Dominancia de Pareto y conjunto óptimo de Pareto
    h) Frente de Pareto
2. Antecedentes Históricos
    a) Orígenes de la optimización multiobjetivo
    b) Clasificación de técnicas
    c) Revisión rápida de enfoques usados en investigación de operaciones
3. Algoritmos Evolutivos
    a) Motivación para resolver problemas multiobjetivo
    b) Técnicas basadas en funciones de agregación (lineales o no lineales)
    c) Técnicas poblacionales
    d) Técnicas basadas en jerarquización de Pareto
    e) Otras técnicas
4. Técnicas para Mantener Diversidad
    a) Nichos y compartición de aptitud
    b) Operadores de agrupamiento (crowding)
    c) Otros esquemas
5. Funciones de Prueba
    a) ¿Cómo diseñarlas adecuadamente?
    b) Ejemplos sin restricciones
    c) Ejemplos con restricciones
    d) Optimización combinatoria
    e) Problemas del mundo real
    f) Problemas que no se han abordado
6. Métricas
    a) ¿Cómo comparar dos algoritmos multiobjetivo?
    b) Cantidad de elementos del conjunto de Pareto
    c) Dispersión
    d) Cercanía al verdadero frente de Pareto

e) Métodos estadísticos
f) Otro tipo de métricas
g) Limitantes de las métricas

7. Teoría
   a) Conjuntos parcialmente ordenados
   b) Convergencia de algoritmos evolutivos multiobjetivo
   c) Nichos y otros métodos para mantener diversidad
   d) Restricciones a la cruza
   e) Análisis de complejidad de los principales algoritmos evolutivos multiobjetivo
   f) Costo computacional
8. Algoritmos Evolutivos Multiobjetivo Paralelos
   a) Filosofía
   b) Paradigmas
   c) Ejemplos
9. Toma de Decisiones Multicriterio
   a) Actitud del tomador de decisiones
   b) Incorporación de preferencias en algoritmos evolutivos multiobjetivo
   c) Puntos a tomar en consideración
10. Otras heurísticas multiobjetivo
   a) Recocido simulado
   b) Búsqueda tabú
   c) La colonia de hormigas
   d) Aprendizaje por refuerzo
   e) Algoritmos meméticos
   f) Optimización mediante cúmulos de partículas
   g) Técnicas adicionales (algoritmos culturales, sistema inmune artificial, búsqueda cooperativa, etc.)
11. Áreas de investigación futura
   a) Toma de decisiones
   b) Nuevos algoritmos
   c) Teoría
   d) Nuevas heurísticas
   e) Nuevas métricas
   f) Búsqueda local
   g) Estructuras de datos espaciales para poblaciones secundarias
   h) Eficiencia
   i) Ideas no exploradas

Bibliografía

1. (Libro de texto): Coello Coello, Carlos A.; Van Veldhuizen, David A. & Lamont, Gary B. “Evolutionary Algorithms for Solving Multi-Objective Problems”, Kluwer Academic Publishers, New York, ISBN 0-3064-6762-3, May 2002.
2. http://delta.cs.cinvestav.mx/ccoello/EMOO

Tópicos selectos de Inteligencia Artificial: Sistemas de Soporte a la Toma de Decisiones

Objetivo

a. Estudio de metodologías y herramientas, matemáticas y computacionales, para el análisis, diseño y desarrollo de sistemas de soporte a la Toma de Decisiones.
b. Estudio de las metodologías y herramientas para el análisis, diseño y desarrollo de los procesos involucrados en la Toma de Decisiones.
c. Estudio de los lenguajes para el análisis, diseño y desarrollo de sistemas de soporte a la Toma de Decisiones.
d. Estudio las técnicas de Inteligencia Artificial, de Teoría de Juegos y de Administración del Conocimiento, con las cuales se fundamenta el desarrollan de sistemas inteligentes para la Toma de Decisiones.

Contenido

1.a.1. Introducción y antecedentes de la Toma de Decisiones
1.a.2. Modelación formal de la Toma de Decisiones
   a. álgebras de conjuntos para la Toma de Decisiones
   b. Lógicas multi-valuadas para la Toma de Decisiones
1.a.3. Análisis y diseño de Ontologías para la Toma de Decisiones
1.a.4. Análisis, diseño y programación de procesos dinámicos para la Toma de Decisiones
1.a.5. Técnicas de Inteligencia Artificial para la Toma de Decisiones
1.a.6. Técnicas de Teoría de Juegos para la Toma de Decisiones
1.a.7. Técnicas de Administración del Conocimiento para la Toma de Decisiones
1.a.8. Desarrollo de casos de estudio
1.a.9. Análisis y diseño de pruebas
1.a.10. Puesta en marcha de sistemas de Toma de Decisiones

Bibliografía

Alain Haurie and Georges Zacour (Editor), Dynamic Games: Theory & Applications, Springer Verlag, 2005, ISBN: 978-0-387-26117-1.
Vincent A.W.J. Marchau, Warren E. Walker, Pieter J.T.M. Bloemen and Steven W. Popper (Editors), Decision Making under Deep Uncertainty: From Theory to Practice, Springer, 2019.

Tópicos Selectos en Redes Neuronales Artificiales

Objetivo

Comprender y aplicar los métodos neuronales a la resolución de problemas complejos inspirándonos en el funcionamiento de nuestro cerebro para su concepción modular, local, distribuida y paralela con el fin de crear sistemas fácilmente adaptables e integrables a otros.

Contenido

1. Neuromimetismo (modelación en neurociencias)
    a) Mecanismos inspirados biológicamente
    b) Sistemas ópticos biológicamente inspirados
    c) Campos receptivos

2. Niveles de percepción
    a) Filtrado de bajo nivel (brillo, iluminación y contraste)
    b) Multi-filtrado espacial (ilusiones ópticas)
    c) Niveles de atención

3. Codificación y decodificación neuronal
    a) El sistema visual
    b) Métodos de correlación inversa: células simples
    c) No linearidades estáticas: células complejas
    d) Campos receptivos en la retina y en el LGN
    e) Construcción de los campos receptivos en V1

4. Neuronas y circuitos neuronales
    a) Modelos de integración y disparo
    b) Conductancias dependientes del voltaje
    c) Modelo de Hodgkin-Huxley
    d) Redes recurrentes
    e) Redes excitatorias-inhibitorias
    f) Redes estocásticas

5. Adaptación y aprendizaje
    a) Reglas de plasticidad sináptica
    b) Modelos causales para la estimación de la densidad

Bibliografía

<u>Libros de base</u>

1. Peter Dayan y L. F. Abbott, *Theoretical neuroscience: Computational and mathematical modelling of neural systems*, 460 pages, MIT Press, 2001, ISBN: 9780262541855
2. G. N. Reeke, R.R. Poznanski, K. A. Lindsay, J.R. Rosenberg y O. Sporns, *Modeling in the Neurosciences: From Biological Systems to Neuromimetic Robotics*, CRC; 2 edition (March 29, 2005), ISBN-10: 0415328683, ISBN-13: 978-0415328685.

3. Yoseph Bar-Cohen, *Biomimetics: Biologically Inspired Technologies*, CRC (November 2, 2005), ISBN-10: 0849331633, ISBN-13: 978-0849331633.
4. Anthony Brabazon y Michael O'Neill, *Biologically Inspired Algorithms for Financial Modelling* (Natural Computing Series), Springer; 1 edition (February 10, 2006), ISBN-10: 3540262520, ISBN-13: 978-3540262527.
5. Eugene M. Izhikevich, *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting*, The MIT Press ISBN 978-0-262-09043-8, 2007
6. *f)* Chris Eliasmith and Charles H. Anderson, *Neural Engineering: Computation, Representation and Dynamics in Neurobiological Systems*, The MIT Press 2003.
7. Bin He, *Neural Engineering*, 488 pages, Springer 1 edition (March 4, 2005), ISBN- 13: 978-0306486098.

Profundizar:

a. Chistof Koch, Biophysics of computation: Information processing in single neurons, 562 pages, Oxford University Press, 1999, ISBN: 9780195181999.
b. Arjen van Ooyen, Modelling Neural Development, A Bradford Book, The MIT Press, USA, 2003, ISBN 0-262-22066-0.
c. Moutier, Sylvain, Inhibition neuronale et cognitive, Lermes Science Publications, Lavoisier, 2003, ISBN 2-7462-0771-0.
d. Klaus Ober-mayer and Terrence J. Sejnowski, Self-Organizing Map Formation: Foundation of Neural Computation, The MIT Press 2001.

Tópicos Selectos en Teoría de Códigos

Objetivo

En la última década hemos presenciado numerosos y significativos avances en la teoría de códigos. El material de este curso se propone motivar el conocimiento de la teoría de códigos, así como presentar algunos de los últimos avances alcanzados en esta disciplina. El curso inicia con una introducción a la teoría de la información de Shannon para después discutir y analizar las propiedades y cotas teóricas de códigos específicos de corrección de error.

Contenido

a) Entropía su caracterización y sus propiedades, códigos Huffman, códigos Shanon-Fano, robustez de las técnicas de codificación, codificación libre de ruido, canal discreto sin memoria y capacidad de canal, teorema fundamental de la teoría de la información.
b) Códigos de corrección de error, principio de la mínima distancia, códigos lineales, cotas de Hamming, cota de Singleton, códigos Reed, códigos BCH, decodificación BCH, decodificación por lista.

Bibliografía

a. F.J. MacWilliams and Neil J.A. Sloane: Theory of Error Correcting Codes. Elsevier/North Holland, Amsterdam, 1981.
b. Vera S. Pless and W. Carry Huffman (Eds.): Handbook of Coding Theory (2 volumes), Elsevier 1998
c. Jacobus H. van Lint: Introduction to Coding Theory. Graduate Texts in Mathematics 86, Springer-Verlag, Berlin, 1999
d. Richard E. Blahut: Theory and practice of error control codes. Addison-Wesley, Reading Massachusetts, 1983.

Tópicos Selectos en Visualización

Objetivo

En este curso se aplicarán las nociones de visión por computadora (VC) en tres dimensiones. La meta de VC es deducir las propiedades y estructura de un mundo tridimensional a partir de una o más vistas bidimensionales. Primero se estudiarán algunas técnicas para procesamiento y análisis de imagen y también se tratarán temas de visualización 3D, animación y realidad virtual, para la creación de modelos tridimensionales y para tener la habilidad de "navegar" a través de ellos. Las herramientas de trabajo serán la librería de procesamiento de imágenes scimagen, y Qt (www.trolltech.com) para el desarrollo de las interfaces gráficas y Mesa (www.mesa3d.org) para interactuar con objetos tridimensionales.

Contenido

1. Introducción al Procesamiento de Imagen
    a) Representación de una Imagen digital
    b) Modelo general para el procesamiento de imágenes
    c) Elementos de un sistema de procesamiento digital de imágenes: adquisición, almacenamiento, una computadora, comunicación, despliegue y software.

2. Fundamentos de Imágenes Digitales
    a) Un modelo simple de imagen
    b) Muestreo y cuantización
    c) Relaciones entre pixeles: vecinos, conectividad, distancia, operaciones aritméticas/lógicas.
    d) Geometría de imágenes: transformaciones y proyecciones.
    e) Filtrado espacial.

3. Segmentación de Imagen
    a) Detección de discontinuidades
    b) Umbralización

4. Representación de la forma y reconocimiento del objeto
    a) Esquemas de representación. El esqueleto de una región. Códigos de cadena.
    b) Descriptores de fronteras: momentos.
    c) Morfología.
    d) Elementos de análisis de imagen
    e) Métodos de decisión teórica: emparejamiento (matching), clasificadores óptimos estadísticos, redes neuronales.

5. Algoritmos geométricos
    a) Algoritmo optimo para encontrar el par de puntos más cercano.
    b) Algoritmos para encontrar la cubierta convexa (convex hull)
    c) La lista de aristas doblemente ligada
    d) Algoritmo optimo para encontrar el diagrama de Voronoi.

6. Reconstrucción del volumen a partir de líneas de contornos
    a) Extracción de las líneas de contorno
    b) Muestreo de las curvas extraídas
    c) Triangulación y visualización.

7. Reconstrucción a partir de dos o más vistas
    a) Calibración de una cámara. Parámetros intrínsecos y extrínsecos
    b) Estimación de los parámetros de una cámara
    c) Detección de esquinas basado en la trasformada de Hough (5)
    d) Triangulación para recobrar la tercera dimensión
    e) Geometría equipolar. El tensor trifocal. Soluciones no-lineales

Bibliografía

a. F.P. Preparata and M.I. Shamos. Computational Geometry, Springer-Verlag. 1985.
b. E. Trucco and A. Verri, Introductory Techniques for 3D Computer Vision 1998, Prentice Hall.
c. R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision 2nd edition, 2003, Cambridge
d. R. Johnsonbaugh. Discrete Mathematics. 4th ed. Prentice Hall. 1997
e. Corner detection based on modified Hough transform F. Shen and H. Wang, Pattern Recognition Letters 23 (2002) 1039-1049
f. R.C. Gonzalez and R.E. Woods, Digital Image Processing 1992, Addison Wesley
g. Abigail Martínez Rivas, Reconstrucción del volumen a partir de su mapa de contornos. Tesis de Maestría. 2005 Sección de Computación, Cinvestav.
h. Ravishankar Chityala and Sridevi Pudipeddi, Image Processing and Acquisition Using Python, CRC Press, 2020, ISBN: 978-0367198084.

Tópicos Selectos en sistemas distribuidos: *eLearning*

Objetivo

Explorar las herramientas de diseño de infraestructura computacional para el aprendizaje asistido por computadora. Técnicas de implementación adecuada y de integración de servicios educativos a distancia

Descripción

Se revisará varias herramientas para la integración de servicios de aprendizaje asistido por computadora y las técnicas más relevantes de cómputo distribuido relativas a este tema.

Contenido

1. Sistemas distribuidos
    a. Técnicas para la distribución de Procesos
    b. Administración de redes locales y de amplio alcance
    c. Cómputo ubicuo
    d. Aspectos de seguridad informática relevantes

2. Diseño de contenido educativo
    a. Diseño y secuenciación de contenido educativo
    b. Metodologías para el aprendizaje por computadora

3. Servicios Web
    a. Servicios web para el aprendizaje a distancia
    b. Características de interconexión abierta y social en la WWW
    c. Formas de aumentar la presencia social en los eventos virtuales de aprendizaje y asíncronos

4. Impactos sociales del aprendizaje asistido por computadora
    a. Impacto en la educación
    b. Impacto en la industria

Bibliografía

- Ruth C. Clark, Richard E. Mayer, "e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning", 4th Edition, ISBN: 978-1-119-15866-0, 2016

- Zongmin Ma, Web-based Intelligent E-learning Systems, Idea Group Inc (IGI), 2006

- Huiwei Wang, Huaqing Li, Bo Zhou, Distributed Optimization, Game and Learning Algorithms, Theory and Applications in Smart Grid Systems, Springer, 2021

Visión

Objetivo

Se revisará la teoría para la reconstrucción tridimensional de escenas a partir de una o varias imágenes bidimensionales, tomadas por una cámara convencional. Se hará énfasis en los métodos para obtener la reconstrucción a partir de las correspondencias de puntos entre las imágenes, lo que se conoce como *autocalibración de la cámara*.

Contenido

1. Geometría proyectiva
2. Métodos numéricos
    a) Propiedades de la descomposición en valores singulares (SVD)
    b) El método de Gauss-Newton
    c) El método Levenberg-Marquardt
    d) Heurísticas para optimización nolineal
3. Modelo para la cámara obscura. Parámetros intrínsecos y extrínsecos de la cámara
4. Autocalibración de la cámara con homografías.
5. Autocalibración de la cámara usando cuboides.
6. La geometría epipolar
7. El tensor trifocal
8. Reconstrucción usando varias cámaras
9. Visualización de la reconstrucción con mapeo de texturas

Bibliografía

1) R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision 2nd edition, 2003, Cambridge
2) E. Trucco and A. Verri, Introductory Techniques for 3D Computer Vision 1998, Prentice Hall.

Tópicos selectos: Algorítmos en Gráficas

Objetivo

Diseñar algoritmos específicamente para trabajar con gráficas y resolver una gran variedad de problemas de manera eficiente.

Aprender algoritmos clásicos y actuales en gráficas.

Descripción

Las gráficas son estructuras matemáticas claves para el análisis de distintos tipos de redes, por ejemplo, las redes sociales, las redes biológicas, Internet, las redes móviles ad hoc, etc. Si bien el área de algoritmos en gráficas es ya un área clásica de la Teoría de la Computación y las Matemáticas Discretas, ésta ha tenido mucha actividad en los últimos años. En este curso se revisará a mayor profundidad el impacto de la Teoría de Gráficas en distintas aplicaciones.

Contenido

1. Introducción al área y un poco de historia
    a. Conceptos básicos, tipos de gráficas
    b. Representación de gráficas y ejemplos
    c. Algoritmo elemental: recorrido
    d. Análisis formal de los algoritmos elementales de recorrido
    e. Software a usar en el curso: Sage Math

2. Extracción de información de estructuras
    a. Web search engines y el algoritmo PageRank
    b. Representación computacional de gráficas
    c. Web crawler
    d. DFS y BFS

4. Algoritmo de Tarjan para encontrar componentes fuertemente conexas
    a. Árboles generadores de peso mínimo y sus propiedades
    b. Algoritmo de Prim-Dijkstra-Jarnik, algoritmo de Boruvka y algoritmo de Kruskal.
    c. Gráficas dirigidas acíclicas y ordenamiento topológico
    d. Caminos más cortos y más largos en gráficas acíclicas dirigidas
    e. Algoritmos de Dijkstra, Bellman-Ford, y Johnson
    f. Algoritmo A*

5. Recorridos en gráficas
    a. Tour de Euler y el problema del agente viajero
    b. Algoritmos de aproximación y tasa de aproximación
    c. Heurística MST-doubling y heurística de Christofides
    d. Un algoritmo de programación dinámica para el problema del agente viajero
    e. Máximo flujo, mínimo corte
    f. Algoritmo de Ford-Fulkerson

6. Emparejamientos, gráficas bipartidas
    a. Algoritmo Hopcroft-Karp
    b. Uso de emparejamientos para encontrar cubiertas máximas y conjuntos independientes, partición en el mínimo número de rectángulos
    c. Emparejamiento estable, algoritmo de Gale-Shapley

7. Problemas complejos en gráficas
    a. Número de Strahler, coloración de gráficas, coloración glotona
    b. Interval graphs y gráficas perfectas
    c. Gráficas cordales y el uso de BFS-lexicográfico para encontrar un orden de eliminación
    d. Métodos para generar redes sintéticas: modelo Erdos-Rényi, ERGM, gráficas aleatorias con grado fijo, Barabási-Albert, y modelo de Kleinberg
    f. Centralidad, degeneración y k-cores
    g. Clanes, cota de Moon-Moser en clanes de tamaño máximo, y algoritmo de Bron-kerbosch

8. Gráficas planares
    a. Road maps, árboles generadores de grids, coloración y el teorema de los cuatro colores
    b. Dualidad, dualidad de Euler, biparticiones y la fórmula de Euler
    c. Verificación de planaridad y el algoritmo basado en ciclos
    d. El teorema de Fáry y bosques de Schnyder

Bibliografía básica

a. Kayhan Erciyes, Guide to Graph Algorithms - Sequential, Parallel and Distributed. Texts in Computer Science, Springer 2018, ISBN 978-3-319-73234-3
b. Shimon Even, Guy Even, Graph Algorithms, Second Edition. Cambridge University Press 2012, ISBN 978-0-521-73653-4
c. Maarten van Steen, Graph Theory and Complex Networks: An Introduction, Maarten van Steen, 2010
d. Vadim Zverovich, Research Topics in Graph Theory and Its Applications 1st Edition, Cambridge Scholars Publishing; 1st edition, 2019

Tópicos selectos de Aprendizaje automático

Descripción

En la era de Big Data, existe una necesidad cada vez mayor de desarrollar e implementar algoritmos que puedan analizar e identificar conexiones en esos datos. El aprendizaje automático es clave para desarrollar sistemas inteligentes y analizar datos en social, ciencia e ingeniería. Esta tecnología tiene numerosas aplicaciones del mundo real que incluyen control robótico, minería de datos, navegación autónoma y bioinformática, redes sociales, negocios, etc. En este curso se presenta una introducción a los modelos y métodos fundamentales del aprendizaje automático moderno. Cubre los conceptos fundamentos, así como los algoritmos esenciales para el aprendizaje supervisado, no supervisado, y por refuerzo.

Contenido

1. Introducción
2. Regresión lineal
    - 2.1. Representación del modelo
    - 2.2. Descenso de gradiente para regresión lineal
    - 2.3. Descenso de gradiente para múltiples variables
3. Regresión logística
    - 3.1. Clasificación
    - 3.2. Regresión logística
4. Redes neuronales artificiales
    - 4.1. Representación del modelo
    - 4.2. Algoritmo retropropagación
5. Máquinas vectoriales de soporte
    - 5.1. Objetivo de optimización
    - 5.2. Kernels
    - 5.3. Clasificación de margen grande
6. Aprendizaje no supervisado
    - 6.1. Aprendizaje no supervisado: Introducción
    - 6.2. Algoritmo de K-means
7. Reducción de dimensionalidad
    - 7.1. Motivación: compresión y visualización de datos
    - 7.2. Análisis de componentes principales (PCA)
8. Aprendizaje profundo
    - 8.1. Long short-term memory (LSTM)
    - 8.2. Redes neuronales convolucionales (CNN)
9. Aprendizaje por refuerzo (Reinforcement Learning)
    - 9.1. Proceso de decisión de Markov (MDP)
    - 9.2. Q-Learning
10. Ejemplo de aplicación
    - 10.1. Aprendizaje con TensorFlow
    - 10.2. Aprendizaje con R Elementos de clasificación

Bibliografía

[1] Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, Fourth edition, Pearson, 2020
[2] Ethem ALPAYDIN, Introduction to Machine Learning, fourth edition, The MIT Press, 2020
[3] Ian Goodfellow, Yoshua Bengio and Aaron Courville, Deep Learning, The MIT Press, 2016
[4] Jeremy Watt, Reza Borhani and Aggelos K. Katsaggelos, Machine Learning Refined: Foundations, Algorithms, and Applications, 2nd Edition, Cambridge University Press, 2020

Tópicos selectos para HPC 1

Objetivo

El alumno aprenderá los conceptos fundamentales de sistemas operativos, procesos, programación multinúcleo, programación paralela y CUDA y arquitectura de procesadores y aceleradores vigentes en la computación de alto Rendimiento.

La currícula cubre tres cursos seriados. El objetivo es aprender los conceptos generales de HPC o Computación de alto Rendimiento, programación concurrente, paralela y distribuida, y herramientas para estos entornos

Contenido:

1.- Introducción
- 1.1 Arquitecturas paralelas
- 1.2 Organizacion de computadoras paralelas
- 1.3 Ciclo de instrucción
- 1.4 Paralelizacion llvm
- 1.5 Arquitecturas muticore
- 1.6 Arquitecturas manycore

2. Conceptos de sistemas operativos
- 2.1 Procesos y calendarizadores
- 2.2 Mecanismos de comunicación
- 2.3 Sistemas de archivos
- 2.4 Concurrecia, multihilos y paralelismo
- 2.5 Modelos de programación

3. Programación multicore
- 3.1 Programación con pthreads
- 3.2 Programación con openmp

4. Diseño de programas paralelos
- 4.1. Metodología de ian foster.
- 4.2. Patrones para programación paralela

5. Análisis y rendimiento de programas paralelos
- 5.1 Tiempo, aceleración y eficiencia de programas paralelos
- 5.2 Ley de amdahl y sus extensiones
- 5.3 Ley de gustafson-barsis y metrica karp-flapp
- 5.4 Análisis de consumo de energía

6. CUDA
- 6.1 Tarjetas aceleradoras gráficas y CUDA
- 6.2 Herramienas de programación
- 6.3 Estructura de programa CUDA
- 6.4 Bloques e hilos
- 6.5 Manejo de datos multidimensionales

6.6 Manejo de memoria y reducción
6.7 Estrategias de optimización
6.8 Extensiones a multiples tarjetas de vídeo

Bibliografía:

[1] Dongarra, Jack, et al. Sourcebook of parallel computing. Vol. 3003. San Francisco: Morgan Kaufmann Publishers, 2003.
[2] Armstrong, Joe. Programming Erlang: software for a concurrent world. Pragmatic bookshelf, 2013.
[3] Mattson, Timothy G., Beverly Sanders, and Berna Massingill. Patterns for parallel programming. Pearson Education, 2004.
[4] Andrews, Gregory R. Foundations of multithreaded, parallel, and distributed programming. Vol. 11. Reading: Addison-Wesley, 2000.
[5] Quinn, J. Michael. Parallel Programming in C with MPI and OpenMP. McGraw Hill. 2004.
[6] Cook, Shane. CUDA programming: a developer's guide to parallel computing with GPUs. Newnes, 2012.
[7] Sanders, J. CUDA by example: an introduction to general-purpose GPU programming. Addison-Wesley Professional. 2010.
[8] Pllana, Sabri and Xhafa, F. Programming multi-core and many-core computing systems. Wiley. 2017.

Tópicos selectos para HPC II

Objetivo

El alumno aprenderá los conceptos fundamentales de Sistemas Distribuidos, Middleware (CORBA y MPI), y programación paralela con paso de mensajes y programación paralela híbrido-heterogénea.

Contenido:

1. Sistemas distribuidos
    - 1.1 Introducción
    - 1.2 Algoritmos distribuidos
    - 1.3 Tiempo lógico
    - 1.4 Sincronización
    - 1.5 Transacciones
    - 1.6 Detección de fin-ejecución

2. Middlewares
    - 2.1 Introducción
    - 2.2 Diseño basado en Middlewares
    - 2.3 Arquitecturas
    - 2.4 Diseño de Api's para Middlewares
    - 2.5 Casos CORBA y MPI

3. CORBA
    - 3.1 Conceptos y elementos delestandar de corba
    - 3.2 El IDL
    - 3.3 El sistema de tiempo de ejecución ORB
    - 3.4 Objetos remotos

4. Programación MPI
    - 4.1 Modelo de programación con paso de mensajes
    - 4.2 Tipos de comunicación
    - 4.3 Manejo de datos

5. Programación paralela híbrida-heterogénea
    - 5.1 introducción
    - 5.2 Diseño de programas
    - 5.3 Programacion paralela híbrida con MPI y Openmp
    - 5.4 Programacion con Openmp y CUDA
    - 5.5 Programación Distribuida y heterogénea con MPI y CUDA
    - 5.6 Programación híbrida.heterogénea con MPI, Openmp y CUDA

Bibliography:

[1] Tanenbaum, Andrew S., and Maarten Van Steen. Distributed systems: principles and paradigms. Prentice-Hall, 2007.
[2] Andrews, Gregory R. Foundations of multithreaded, parallel, and distributed programming. Vol. 11. Reading: Addison-Wesley, 2000.

[3] Buyya, Rajkumar, James Broberg, and Andrzej M. Goscinski, eds. Cloud computing: Principles and paradigms. Vol. 87. John Wiley & Sons, 2010.
[4] Verissimo, Paulo, and Luis Rodrigues. Distributed systems for system architects. Vol. 1. Springer Science & Business Media, 2012.
[5] Cerami, Ethan. Web services essentials: distributed applications with XML-RPC, SOAP, UDDI & WSDL. O' Reilly Media, Inc., 2002.
[6] Laurent, Simon St, et al. Programming Web Services with XML-RPC: Creating Web Application Gateways. O' Reilly Media, Inc., 2001.
[7] Etzkorn, Letha Hughes. Introduction to Middleware: Web Services, Object Components, and Cloud Computing. CRC Press, 2017.
[8] Britton, Chris, and Peter Bye. IT architectures and middleware: strategies for building large, integrated systems. Pearson Education, 2004.
[9] Quinn, J. Michael. Parallel Programming in C with MPI and OpenMP. McGraw Hill. 2004.

Tópicos selectos para HPC III

Objetivo

El alumno aprenderá conceptos Avanzados de HPC: Herramientas para BigData, Cómputo en la Nube, Cómputo en malla y diseño de herramientas e infraestructura para centros de cómputo de alto rendimiento y centro de datos.

Contenido:

1) Diseño de herramientas para Sistemas Distribuidos

1.1 Patrones de Middlewares
1.2. Arquitecturas de Cloud Computing
1.3 Arquitecturas de Grid Computing
1.4 Infraestructura como servicios

2) Herramientas para Big Data

2.1. Hadoop
2.2. Storm
2.3. Spark
2.4. Cassandra

3) Herramientas de Soporte para HPC

3.1. Programación Orientada a tareas
3.2. Herramientas para Sistemas Híbridos Heterogéneos
3.3. Despachadores para Sistemas heterogéneos
3.4. Manejadores de Colas
3.5. Sistemas de Archivos Paralelos

4) Computación Verde

4.1. Introducción
4.1. Infraestructura de Cómputo: Centro de Datos y Supercómputo
4.2. Indicadores de Energía en Servidores
4.3. Indicadores de Energía para Centro de Datos
4.4. Indicadores de Energía para Centros de Supercómputo

Bibliography:

[1] Buyya, Rajkumar, James Broberg, and Andrzej M. Goscinski, eds. Cloud computing: Principles and paradigms. Vol. 87. John Wiley & Sons, 2010.
[2] Tanenbaum, Andrew S., and Maarten Van Steen. Distributed systems: principles and paradigms. Prentice-Hall, 2007.
[3] Verissimo, Paulo, and Luis Rodrigues. Distributed systems for system architects. Vol. 1. Springer Science & Business Media, 2012.
[4] Cerami, Ethan. Web services essentials: distributed applications with XML-RPC, SOAP, UDDI & WSDL. O'Reilly Media, Inc., 2002.
[5] Laurent, Simon St, et al. Programming Web Services with XML-RPC: Creating Web Application

Gateways. O'Reilly Media, Inc., 2001.
[6] Britton, Chris, and Peter Bye. IT architectures and middleware: strategies for building large, integrated systems. Pearson Education, 2004.
[7] Dulhare, Uma N., Khaleel Ahmad, and Khairol Amali Bin Ahmad, eds. Machine Learning and Big Data: Concepts, Algorithms, Tools and Applications. John Wiley & Sons, 2020.
[8] Salloum, Salman, et al. Big data analytics on Apache Spark. International Journal of Data Science and Analytics 1.3-4 (2016): 145-164.
[9] Iqbal, Muhammad Hussain, and Tariq Rahim Soomro. Big data analysis: Apache storm perspective. International journal of computer trends and technology 19.1 (2015): 9-14.
[10] Jain, Ankit, and Anand Nalya. Learning storm. Packt Publishing, 2014.
[11] Shoro, Abdul Ghaffar, and Tariq Rahim Soomro. Big data analysis: Apache spark perspective. Global Journal of Computer Science and Technology (2015).
[12] Karau, Holden, and Rachel Warren. High performance Spark: best practices for scaling and optimizing Apache Spark. O'Reilly Media, Inc., 2017.
[13] Chebotko, Artem, Andrey Kashlev, and Shiyong Lu. A big data modeling methodology for Apache Cassandra. 2015 IEEE International Congress on Big Data. IEEE, 2015.
[14] Reese, George. Cloud application architectures: building applications and infrastructure in the cloud. O' Reilly Media, Inc., 2009.
[15] Quintero, Dino, et al. Implementing the IBM General Parallel File System (GPFS) in a Cross Platform Environment. IBM Redbooks, 2011.
[16] Pllana, Sabri and Xhafa, F. Programming multi-core and many-core computing systems. Wiley. 2017.
[17] Hu, Wen-Chen, ed. Sustainable ICTs and management systems for green computing. IGI Global, 2012.
[18] Smith, Bud E. Green Computing: Tools and Techniques for Saving Energy, Money, and Resources. CRC Press, 2013.
[19] Feng, Wu-chun, ed. The Green Computing Book: Tackling Energy Efficiency at Large Scale. CRC Press, 2014.
[20] Khosrow-Pur, M. Green Computing Strategies for Competitive Advantage and Business Sustainability (Advances in Systems Analysis, Software Engineering, and High Performance Computing).

Análisis Estadísticos de Datos

Descripción:

Las aplicaciones de minería de datos y "big data" ocupan cada vez más un lugar central en nuestra sociedad moderna impulsada por el conocimiento; respaldada por avances en el poder de la computación, la adquisición automatizada de datos, el desarrollo de redes sociales y el software de Internet interactivo y ligado. Es una introducción completa a los métodos estadísticos para la minería de datos y el descubrimiento de conocimientos. También puede servir como base de la informática estadística y el análisis de datos. Está diseñado para los estudiantes que necesitan aplicar el aprendizaje estadístico a sus datos modernos en sus proyectos. Cubre una introducción técnica al aprendizaje estadístico moderno y al análisis de datos. Se ilustrarán amplios ejemplos mediante el análisis de conjuntos de datos en genómica, biomedicina, teledetección ecológica, astronomía, socio economía, marketing, publicidad y finanzas, entre muchos otros.

Contenido:

1. Introducción: análisis de datos y minería de datos
2. Probabilidad básica y distribuciones estadísticas
3. Manipulación de datos
4. Visualización de datos y gráficos estadísticos
5. Inferencia estadística
6. Aprendizaje supervisado--Regresión
7. Aprendizaje supervisado--Clasificación
8. Aprendizaje no supervisado

Lenguaje que ocupa: R

Bibliography:

[1] Walter W. Piegorsch, Statistical Data Analytics: Foundations for Data Mining, Informatics, and Knowledge Discovery, Wiley, 2015 (Textbook)
[2] Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani, An Introduction to Statistical Learning: with Applications in R, Springer, 2013
[3] Trevor Hastie, Robert Tibshirani, Jerome Friedman, The Elements of Statistical Learning, Springer, 2001
[4] Kevin P. Murphy. Machine learning: a probabilistic perspective, MIT Press, 2012.
[5] Christopher M. Bishop. Pattern recognition and machine learning, Springer, 2009.

Introducción al Cómputo Reconfigurable

Objetivo

Se presentan los elementos básicos para crear arquitecturas y algoritmos que utilicen dispositivos programables.

Descripción

En este curso se utiliza el paradigma de los dispositivos programables FPGAs para implementar en ellos algoritmos en hardware o en hardware/software que tengan un mejor rendimiento que las implementaciones en software puras. Se revisan las características de las aplicaciones susceptibles de tener mejoras mediante dicho paradigma y se presentaciones algunas soluciones como casos de estudio. Se revisan también algunas estrategias para diseñar arquitecturas con procesadores reconfigurables acoplados.

Contenido

1. Introducción a la lógica digital.
    a. Diferencia entre sistemas digitales y analógicos.
    b. Sistemas numéricos de notación posicional.
    c. Códigos binarios.
    d. Elementos básicos de lógica y tablas de verdad.
    e. Circuitos lógicos.

2. Introducción a FPGA.
    a. Programación de los FPGAs
    b. Arquitectura de los FPGAs
    c. Configuración y elementos de ruteo
    d. Recursos disponibles en un FPGA de Xilinx
    e. Elementos de reloj
    f. Familias de FPGA de Xilinx

3. Implementación de diseños digitales en FPGA.
    a. Introducción al VHDL
    b. Elementos básicos del lenguaje
    c. Entidad, arquitectura y tipo de datos en VHDL
    d. Asignaciones a señales
    e. Simulación de comportamiento
    f. Diseño jerárquico basado en componentes

4. Álgebra de boole y técnicas de simplificación
    a. Funciones de conmutación y formas canónicas; expansiones en min-términos y max-términos.
    b. Simplificación de funciones Booleanas.
        i. Mapas de Karnaugh.

5. Elementos de lógica combinatoria
    a. Decodificadores.

b. Multiplexores.
c. Comparadores.
d. Sumador y sustractor.

6. Elementos avanzados de los FPGAs
   a. Memorias de solo lectura (ROM).
   b. Arreglos lógicos programables.
      i. Bloques en FPGA
      ii. Bloques de memoria
   c. Multiplicadores y bloques de DSP
   d. Sistemas secuenciales simples (SSS).
   1) Flip-Flops.
   2) Descripción del funcionamiento de un SSS.
   3) Diagramas de tiempo.
   4) Diagramas de estado.
   5) Registros y contadores.
   6) Memoria de acceso aleatorio (RAM).

7. Implementación de diseños digitales en FPGA.
   a. Componentes secuenciales
      i. Procesos combinacionales y secuenciales
      ii. Condicionales
      iii. Declaración case
   b. Diseño jerárquico basado en componentes
      i. Conexión de componentes
      ii. Generalización de componentes
      iii. Duplicación de componentes
   c. Máquinas de estado finito con VHDL
      i. Asignación de estados
      ii. Selección de elementos secuenciales
      iii. Síntesis de un SSS definido por diagramas de estado

8. Sistemas Programables en Chip (SoPC).
   a. Sistema empotrado SoC
   b. Arquitectura conceptual de Zynq de Xilinx
   c. Flujo de diseño con SoC
   d. Aplicaciones reales.
   e. Arquitectura del Zynq 7000
   f. Flujo de diseño en Vivado
   g. Construcción de un sistema empotrado en Zynq
      a. Interfaz de periféricos AXI
      b. Transferencia de Datos con DMA

9. Sistemas operativos de tiempo real
   a. FreeRTOS

Bibliografía

1. Floyd, Thomas L., Fundamentos de Sistemas Digitales. Prentice Hall. 9a. Edición, 2006.
2. Hayes, John P., Introduction to Digital Logic Design. Addison Wesley, 1993.
3. Katz, R. y Borriello, G., Contemporary Logic Design. Addison Wesley, 2005.
4. Morris Mano, M., Diseño Digital. Prentice Hall. Tercera Edición, 2003, (incluye CD-ROM).
5. Morris Mano, M., Kimer, Charles R., Fundamentos de Diseño Lógico y Computadoras. Prentice Hall, 2005.
6. Wackerly, J. F., Digital Design Principles and Practices. Prentice Hall, 2006.
7. HORIE Tetsuya, How to make RISC-V Microcomputer using FPGA for programmer, NextPublishing Authors Pres, 2020
8. Vaibbhav Taraate, Logic Synthesis and SOC Prototyping, Springer, 2020.
9. Richard E. Haskell, Darrin M. Hanna, Introduction to Digital Design Using Digilent FPGA Boards, LBE Books; 1st Edición, 2019.
10. Iouliia Skliarova, Valery Sklyarov, FPGA-BASED Hardware Accelerators, Springer, 2019.
11. Alexander Barkalov, Larysa Titarenko, Małgorzata Mazurkiewicz, Foundations of Embedded Systems, Springer, 2020
12. Felipe Neves, Hands-On Embedded System Design, Packt, 2018.
13. Lennart Lindh, Tommy Klevin, Advanced HW/SW Embedded System for Designers, Independently published, 2018

Tópicos Selectos: Códigos lineales

Objetivos.
Al terminar el curso el alumno sera Capaz de manejar los parámetros de un código lineal, como influyen en el código y el equilibrio entre estos; estudiar´a y analizar´a los principales códigos lineales utilizados en la actualidad, entendiendo la relevancia, sus diferencias y distintas capacidades. De importancia es el análisis de las distintas técnicas de codificación y decodificación, de modo que se comprendan las ventajas y desventajas de cada una de ellas. Adicionalmente, los alumnos realizaran la implementación computacional de éstas. Se llevará a cabo un proyecto de investigación con base en lo aprendido. Por ejemplo, un análisis profundo y manejo de algún código no considerado en el salón de clases, la distribución de pesos de un código lineal, funciones booleanas y los códigos, la criptografía y los códigos línea leales, códigos y diseños, códigos y esquemas de compartición de secretos, etc.

Contenido:

1. Códigos
2. Códigos lineales
3. Códigos de Hamming
4. Construcción de nuevos códigos lineales a partir de otros
5. Código simplex
6. Códigos MDS
7. Funciones booleanas
8. Códigos de Reed Muller
9. Código de Golay
10. Campos finitos
11. Códigos cíclicos
12. Códigos BCH
13. El criptosistema McEliece
14. Realización de proyectos

Bibliography.

1. F. J. MacWilliams; N. J. A. Sloane. The Theory of Error-Correcting Codes, Elsevier Science Publishers. ISBN: 0 444 85193 3
2. S. Roman. Coding and Information Theory, Springer.
3. R. J. McEliece. The Theory of Information and Coding, M. ISBN: 0 521 00095 5
4. Cunsheng, Ding. Deigns from Linear Codes. https://doi.org/10.1142/11101
5. Betten, A.; Braun, M.; Fripertinger, H.; Kerber, A.; Kohnert, A.; Wassermann, A.
6. Error-Correcting Linear Codes. Springer.
7. Lidl, Rudolf; Niederreiter, Harald. Finite Fields.
8. Hoffman, Kenneth; Kunze, Ray. Linear Algebra.
9. Kenneth Ireland; Michael Rosen. A Classical Introduction to Modern Number Theory, Springer.

Tópicos Selectos en Minerias de Datos Avanzada

Objetivo

The course is designed especially for computer science postgraduate students who need text mining and web mining techniques directly or indirectly in their thesis work. Basic concepts and the state-of-art techniques and representative algorithms will be introduced.

Description:
With the rapid advance of computer and internet technologies, a plethora of data accumulates. Data will not turn into knowledge no matter how long it is kept. Mining nuggets from data will add values to what we are currently doing in many areas. Data mining is a process that finds the valuables among the mountains of data.

We will review and examine the present techniques and the theories behind them and explore new and improved techniques for real world data mining applications. The arrangement of the course will encourage active class participation, creative thinking, and hands-on project development among the participants. Several course projects on some specific aspect of this emerging field will be required for each student to explore some in-depth issue(s) and gain data mining experience.

Contenido:

1. Information retrieval
2. Text representation and search
3. Representative techniques of text processing
4. Evaluation techniques
5. Web Mining
6. Social Network Analysis
7. Web Crawling
8. Opinion Mining and Sentiment Analysis
9. Web Usage Mining

Required text:

1.- Ricardo Baeza-Yates, Berthier Ribeiro-Neto (2010). Modern Information Retrieval. The Concepts and Technology behind Search. Second edition. Addison-Wesley.
2.- Bing Liu (2011). Web Data Mining. Exploring Hyperlinks, Contents, and Usage Data. Springer. ISBN: 978-3-642-19459-7.
3.-Pang-Ning Tan, Michael Steinbach, and Vipin Kumar (2005) Introduction to Data Mining, Addison Wesley http://www-users.cs.umn.edu/~kumar/dmbook

Tópicos Selectos de Aprendizaje Profundo

Objetivo

Los temas de aplicación tratados en el curso incluyen clasificación de imágenes, previsión de series temporales, vectorización de texto (tf-idf y word2vec), traducción de lenguaje natural, reconocimiento de voz y aprendizaje de refuerzo profundo. Los estudiantes aprenderán a utilizar MATLAB Deep Learning ToolboxTM para crear una variedad de redes neuronales profundas: red neuronal convolucional (CNN), red neuronal recurrente (RNN), long short-term memory (LSTM), aprendizaje profundo por refuerzo, etc.

Descripción

Aprendizaje profundo (Deep learning) es una de las habilidades más buscadas en Inteligencia Artificial. Además de los fundamentos teóricos de las redes neuronales, incluyendo la retro propagación y el descenso estocástico del gradiente, los estudiantes obtendrán conocimientos fundamentales de algoritmos de aprendizaje profundo que se desarrollan para extraer representaciones de características de alto nivel de datos, y obtendrán experiencia práctica en la construcción de redes neuronales.

Contenido:

1. Redes neuronales y redes de funciones de base radial (RBF)
2. Aprendizaje automático de redes neuronales poco profundas
3. Asuntos para generalizar aprendizajes profundos
4. Aprendizaje de redes neuronales profundas
5. Aprendizaje permanente y no supervisado
6. Redes neuronales recurrentes (LSTM)
7. Redes neuronales convolucionales (CNN)
8. Aprendizaje por refuerzo profundo (RL)
9. Temas avanzados en el aprendizaje profundo

Bibliography

1.- Charu C. Aggarwal, Neural Networks and Deep Learning, Springer, September 2018
2.- Ian Goodfellow, Yoshua Bengio and Aaron Courville, Deep Learning, The MIT Press, 2016
3.- Chris A. Mattmann, Machine Learning with TensorFlow, Manning Publications; 2nd ed. (2 Febrero 2021)
4.- John D. Kelleher, Deep Learning, MIT Press (2019)
5.- Richard S. Sutton, Andrew G Barto, Reinforcement Learning: An Introduction, 2nd ed., Bradford Books, 2018

Tópicos Selectos: Interfaces Conscientes del Contexto

Objetivo

Una interfaz de usuario consciente del contexto es aquella que es capaz de identificar el entorno del usuario, con el objetivo de adaptarse y ofrecer un mejor servicio. Los tres elementos más importantes del contexto son: quién se es, con quién se está, y cuáles son los recursos disponibles en el ambiente. El contexto involucra más que la ubicación del usuario, pues existen otros elementos que están en constante movimiento y cambio. El contexto puede incluir elementos como la iluminación, el nivel de ruido, disponibilidad de red, costos de comunicación, ancho de banda, e inclusive situaciones sociales (e.g., si se está con el jefe o con un compañero). De esta manera, el objetivo de éste curso es investigar y analizar los diversos retos que el contexto presenta a las aplicaciones; cómo un sistema computacional puede ser consciente del contexto, cuáles son los mecanismos que permiten reaccionar al cambio de contexto, arquitecturas propuestas, y problemas abiertos.

Temario

1. El ambiente y el contexto
    1. Canales de entrada y salida
    2. Las dimensiones del software consciente de contexto

2.- Reconfiguración contextual manual
    2.1. Información contextual y comandos
    2.2. Acciones reaccionadas por el contexto

3.- Reconfiguración contextual automática
    3.1. Información contextual y comandos
    3.2. Acciones reaccionadas por el contexto

4.- Plasticidad
    4.1. Usabilidad y carga cognitiva
    4.2. Modelos ontológicos
    4.3. Modelos arquetípicos
    4.4. Pasos para la adaptación

5.- Contexto en dispositivos móviles
    5.1. Tipos de métodos
    5.2. Sensores
    5.3. Internet de las cosas

6.- Infraestructura
    6.1. Independencia del hardware, sistema operativo y lenguaje de programación
    6.2. Capacidades mejoradas para el mantenimiento y evolución
    6.3. Compartir sensores, poder de procesamiento, datos y servicios
    6.4. Retos para construir una infraestructura consciente de contexto
        6.4.1 Definición de formatos de datos y protocolos
        6.4.2. Construcción de servicios básicos de infraestructura
        6.4.3 Repartición de responsabilidades
        6.4.4. Alcance y acceso a los datos de contexto

6.4.5. Escalabilidad

Bibliografía recomendada

1. ABDELLATIF, A.A., MOHAMED, A., CHIASSERINI, C.F., TLILI, M.M AND ERBAD, A. Edge computing for smart health: Context-aware approaches, opportunities, and challenges. IEEE Network 33,3 (2019), 196-203.

2. ABOROKBAH, M.M., AL-MUTAIRI, S., SANGAIAH, A.K., AND SAMUEL, O.W. Adaptive context aware decision computing paradigm for intensive health care delivery in smart cities – a case analysis. Sustainable cities and society 41 (2018), 919-924.

3. ABOWD, G.D., DEY, A.K., BROWN, P.J., DAVIES, N., SMITH, M., AND STEGGLES, P. Towards a better understanding of context and context-awareness. In International symposium on handheld and ubiquitous computing (1999), Springer, pp. 304-307

4. BANSAL, M., CHANA, I., AND CLARKE, S. Enablement of iot based context-aware smart home with fog computing. In fog Computing: Breakthroughs in Research and Practice. IGI Global, 2018, pp. 251-263

5. BURNY, N., AND VANDERDONCKT, J. UILAB, a workbench for conducting and reproducing experiments in gui visual design. Proc. ACM Hum.- Comput. Interact. 5, EICS (May 2021)

6. CALVARY, G., COUTAZ, J., AND THEVENIN, D. Supporting context changes for plastic user interfaces: a process and a mechanism. In People and Computers XV – Interaction without Frontiers. Springers, 2001, pp. 349-363.

7. CALVARY, G., COUTAZ, J., THEVENIN, D., LIMBOURG, Q., BOUILLON, L., AND VANDERDONCKT, J. A unifying reference framework for multi-target user interfaces. Interacting with computers 15, 3. (2003), 289-308.

8. CHEN, G., AND KOTZ, D. A survey of context-aware mobile computing research. Dartmouth Computer Science Technical Report TR2000-381 (2000).

9. COUTAZ, J. User interface plasticity: Model driven engineering to the limit! In Proceedings of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems (2010), ACM, pp. 1-8.

(10) DEY, A.K. Understanding and using context. Personal and ubiquitous computing 5, 1 (2001)

(11) DOURISH, P. Seeking a foundation for context-aware computing. Human-Computer Interaction 16, 2-4 (2001), 229-241.

(12) EL-DIN, D.M., HASSANEIN, A. E., AND HASSANIEN, E.E. A Proposed Context-Awareness Taxonomy for Multi-data Fusion in Smart Environments: Types, Properties, and Challenges. Springer International Publishing, Cham, 2021, pp. 511-536.

(13) GAJJAR, M. J. Mobile Sensors and Context-Aware Computing. Morgan Kaufmann, 2017.

(14) GURCAN, F., CAGILTAY, N. E. AND CAGILTAY, K. Mapping Human-Computer interaction research themes and trends from its existence to today: A topic modeling-based Review of past 60 years. International Journal of Human-Computer Interaction 37, 3 (2021), 267-280

(15) HONG, J. I. AND LANDAY, J.A. An infraestructure approach to context-aware computing. Human-Computer Interaction 16, 2-4 (2001), 287-303.

(16) JOSEPH, A. W., VAIZ, J. S., AND MURUGESH, R. Modeling cognitive load in mobile human computer interaction using eye tracking metrics. In Advances in Artificial Intelligence, Software and Systems Engineering (Cham, 2021), T. Z. Ahram, W. Karwowski, and J. Kalra, Eds., Springer International Publishing, pp. 99-106.

(17) LIANG, C. A., MUNSON, S. A., AND KIENTZ, J.A. Embracing four tensions in human-computer interaction research with marginalized people. ACM Trans. Computa.- Hum. Interact. 28,2 (Apr. 2021).

(18) LIU, L. The artistic design of user interaction experience for mobile systems based on context-awareness and machine learning. Neural Computing and Applications (Jun 2021).

[21] MacKenzie, I. S. Human-Computer Interaction: An Empirical Research Perspective, 1st ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2013.

[22] MENDOZA, S., HERNANDEZ-LEÓN, M., SÁNCHEZ-ADAME, L. M., RODRÍGUEZ, J., DECOUCHANT, D., AND MENESES-VIVEROS, A. Supporting student-teacher interaction through a chatbot. In Learning and Collaboration Technologies. Human and Technology Ecosystems (Cham, 2020), P. Zaphiris and A. Ioannou, Eds., Springer International Publishing, pp. 93– 107. 3

[23] Motti, V. G., and Vanderdonckt, J. A computational framework for context-aware adaptation of user interfaces. In IEEE 7th International Conference on Research Challenges in Information Science (RCIS) (May 2013), pp. 1–12.

[24] Murad, C., Munteanu, C., R. Cowan, B., and Clark, L. Finding a new voice: Transitioning designers from gui to vui design. In CUI 2021 - 3rd Conference on Conversational User Interfaces (New York, NY, USA, 2021), CUI '21, Association for Computing Machinery.

[25] Oprea, E. M., Moisescu, M. A., and Caramihai, S. I. Context awareness in enterprise systems design. In 2021 23rd International Conference on Control Systems and Computer Science (CSCS) (2021), pp. 280–286.

[26] Orso, V., Ver`ı, D., Minato, R., Sperduti, A., and Gamberini, L. Are professional kitchens ready for dummies? a comparative usability evaluation between expert and non-expert users. In Human-Computer Interaction. Design and User Experience Case Studies (Cham, 2021), M. Kurosu, Ed., Springer International Publishing, pp. 418–428.

[27] Ortiz, G. Adaptive Web Services for Modular and Reusable Software Development: Tactics and Solutions: Tactics and Solutions. Advances in Web Technologies and Engineering (2328- 2762).

Information Science Reference, 2012.

[28] Perera, C., Zaslavsky, A., Christen, P., and Georgakopoulos, D. Context aware computing for the internet of things: A survey. IEEE communications surveys & tutorials 16, 1 (2013), 414–454.

[29] Schilit, B., Adams, N., and Want, R. Context-aware computing applications. In 1994 First Workshop on Mobile Computing Systems and Applications (Dec 1994), pp. 85–90.

[30] Schilit, B. N., Adams, N., Want, R., et al. Context-aware computing applications. Xerox Corporation, Palo Alto Research Center, 1994.

[31] SCHMIDT, A., ALT, F., AND MÄKELÄ, V. ¨ Evaluation in Human-Computer Interaction – Beyond Lab Studies. Association for Computing Machinery, New York, NY, USA, 2021.

[32] Shishkov, B., and van Sinderen, M. Towards well-founded and richer context-awareness conceptual models. In Business Modeling and Software Design (Cham, 2021), B. Shishkov, Ed., Springer International Publishing, pp. 118–132.

[33] Shneiderman, B. The new ABCs of research: Achieving breakthrough collaborations. Oxford University Press, 2016.

[34] SÁNCHEZ-ADAME, L. M., MENDOZA, S., URQUIZA, J., RODRÍGUEZ, J., AND MENESES-VIVEROS, A. Towards a set of heuristics for evaluating chatbots. IEEE Latin America Transactions 19, 12 (May 2021), 2037–2045.

[35] SÁNCHEZ-ADAME, L. M., URQUIZA-YLLESCAS, J. F., AND MENDOZA, S. Measuring anticipated and episodic ux of tasks in social networks. Applied Sciences 10, 22 (Nov 2020), 8199.

[36] Sottet, J.-S., Ganneau, V., Calvary, G., Coutaz, J., Demeure, A., Favre, J.-M., and Demumieux, R. Model-driven adaptation for plastic user interfaces. In IFIP Conference on Human-Computer Interaction (2007), Springer, pp. 397–410. 4

[37] Temdee, P., and Prasad, R. Context-aware communication and computing: Applications for smart environment. Springer, 2018.

[38] Tidwell, J. Designing interfaces: Patterns for effective interaction design. O'Reilly, 2010.

[39] Triberti, S., Di Natale, A. F., and Gaggioli, A. Flowing Technologies: The Role of Flow and Related Constructs in Human-Computer Interaction. Springer International Publishing, Cham, 2021, pp. 393–416.

[40] VASSEUR, A., LÉGER, P.-M., COURTEMANCHE, F., LABONTE-LEMOYNE, E., GEORGES, V., VALIQUETTE, A., BRIEUGNE, D., RUCCO, E., COURSARIS, C., FREDETTE, M., AND SÉNÉCAL, S. Distributed remote psychophysiological data collection for ux evaluation: A pilot project. In Human-Computer Interaction. Theory, Methods and Tools (Cham, 2021), M. Kurosu, Ed., Springer International Publishing, pp. 255–267.

Tópicos Selectos de Métodos de Evaluación de Interfaces de Usuario

Objetivo

Investigar y analizar los diversos métodos de evaluación de interfaces de usuario existentes en el estado del arte, poniendo especial énfasis en temas de usabilidad y experiencia de usuario. En este contexto, la experiencia de usuario se refiere al estudio e interpretación de las percepciones de los usuarios en eventos específicos de interacción con un sistema. Mientras que usabilidad se refiere al concepto principal de la experiencia de usuario, i.e., la eficacia, eficiencia, y satisfacción que un sistema ofrece para alcanzar un objetivo determinado. De esta manera, ambos términos serán el punto en común de todos los métodos de evaluación que se estudiarán. Esta especialización de la Interacción Humano-Computadora (HCI por sus siglas en inglés), ha permitido a los investigadores tanto en el campo académico como en la industria, mejorar la calidad de las interfaces gráficas de usuario (GUI por sus siglas en inglés), crear experiencias positivas que aumentan la productividad y hacer que los ambientes interactivos evolucionen a la par de las necesidades humanas.

Temario

1.- Contexto histórico
- 1.1. Inicio y crecimiento del HCI y las GUI.
  - 1.1.1. Pioneros de la investigación (e.g., Sutherland, Engelbart y Xerox).
  - 1.1.2 SIGCHI.
  - 1.1.3 La psicología del HCI.
  - 1.1.4. Primeras GUI.
- 1.2. Importancia de la evaluación de interfaces.
  - 1.2.1 Beneficios para el usuario, desarrollador e inversionista.
  - 1.2.2 Intuición y realidad.

2.- Midiendo al humano
- 2.1. Tiempos de reacción y factores humanos.
  - 2.1.1 Sentidos (visión, audición, tacto, gusto y olfato).
    - 2.1.2. Métodos y herramientas de medición (e.g., eye tracking).
  - 2.2.2. El cerebro (percepción, cognición, y memoria).
    - 2.2.3. Métodos y herramientas de medición (e.g., cuestionarios de carga cognitiva).
  - 2.3.3 Análisis, interpretación y presentación de resultados.
    - 2.3.1 Resultados cuantitativos.
    - 2.3.2.
    - 2.3.2. Resultados cualitativos.

3.- Elementos de interacción
- 3.1. Controles “duros” y “suaves”.
  - 3.1.1 Interacción con hardware.
  - 3.1.2 Ergonomía y diseño industrial.
- 3.2. Interacción con GUI.
- 3.2.1. Heurísticas de diseño y usabilidad.
  - 3.2.2. Espacio y tiempo de interacción.
- 3.3. Relación entre controles y pantallas.
  - 3.3.1. Contexto móvil.
  - 3.3.2. Representación de la información.

3.4. Interacción social.

4.- Métodos de Evaluación
- 4.1. Tipos de Métodos.
  - 4.1.1. Estudios de campo.
  - 4.1.2 Estudios en laboratorio.
  - 4.1.3. Estudios en líınea.
  - 4.1.4. Cuestionarios y escalas.
- 4.2. Fases de desarrollo.
  - 4.2.1 Escenarios, storyboards, y sketches.
  - 4.2.2. Prototipos iniciales y rápidos.
  - 4.2.3. Prototipos funcionales.
  - 4.2.4. Productos en el mercado.
- 4.3. Periodo de experiencia a estudiar.
  - 4.3.1. Antes del uso.
  - 4.3.2. “Instantáneas” durante el uso.
  - 4.3.3. Después del uso, experiencias de tarea.
  - 4.3.4. Evaluación a largo plazo.
- 4.4. Evaluador/Proveedor de información.
  - 4.4.1. Expertos.
  - 4.4.2. Un usuario a la vez.
  - 4.4.3. Grupos de usuarios.
  - 4.4.4 Pares de usuarios.

Bibliografía

[1] Albert, W., and Tullis, T. Measuring the user experience: collecting, analyzing, and presenting usability metrics. Newnes, 2013.

[2] Barnum, C. M. Usability testing essentials: ready, set... test! Elsevier, 2010.

[3] Burny, N., and Vanderdonckt, J. Uilab, a workbench for conducting and reproducing experiments in gui visual design. Proc. ACM Hum.-Comput. Interact. 5, EICS (May 2021).

[4] DUMAS, J. S., DUMAS, J. S., AND REDISH, J. A practical guide to usability testing. Intellect books, 1999.

[5] Garrett, J. J. Elements of user experience, the: user-centered design for the web and beyond. Pearson Education, 2010.

[6] Geisen, E., and Romano Bergstrom, J. Chapter 1 - usability and usability testing. In Usability Testing for Survey Research, E. Geisen and J. Romano Bergstrom, Eds. Morgan Kaufmann, Boston, 2017, pp. 1–19.

[7] Grudin, J. From tool to partner: The evolution of human-computer interaction. In Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems (Montreal QC, Canada, 2018), CHI EA ’18, Association for Computing Machinery, p. 1–3.

[8] GURCAN, F., CAGILTAY, N. E., AND CAGILTAY, K. Mapping human–computer interaction research themes and trends from its existence to today: A topic modeling-based review of past 60 years. International Journal of Human–Computer Interaction 37, 3 (2021), 267–280.

[9] Joseph, A. W., Vaiz, J. S., and Murugesh, R. Modeling cognitive load in mobile human computer interaction using eye tracking metrics. In Advances in Artificial Intelligence, Software and Systems Engineering (Cham, 2021), T. Z. Ahram, W. Karwowski, and J. Kalra, Eds., Springer International Publishing, pp. 99–106.

[10] Krug, S. Don't make me think!: a common sense approach to Web usability. Pearson Education India, 2000.

[11] Lazar, J., Feng, J. H., and Hochheiser, H. Research methods in human-computer interaction. Morgan Kaufmann, 2017.

[12] Liang, C. A., Munson, S. A., and Kientz, J. A. Embracing four tensions in humancomputer interaction research with marginalized people. ACM Trans. Comput.-Hum. Interact. 28, 2 (Apr. 2021).

[13] Love, S. Understanding mobile human-computer interaction, 1st ed. Butterworth-Heinemann, 2005.

[14] MacKenzie, I. S. Human-Computer Interaction: An Empirical Research Perspective, 1st ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2013.

[15] Marshall, J. Ux: do less, but with feeling, Oct 2019. 3

[16] MENDOZA, S., HERNANDEZ-LEÓN, M., SÁNCHEZ-ADAME, L. M., RODRÍGUEZ, J., DECOUCHANT, D., AND MENESES-VIVEROS, A. Supporting student-teacher interaction through a chatbot. In Learning and Collaboration Technologies. Human and Technology Ecosystems (Cham, 2020), P. Zaphiris and A. Ioannou, Eds., Springer International Publishing, pp. 93– 107.

[17] Murad, C., Munteanu, C., R. Cowan, B., and Clark, L. Finding a new voice: Transitioning designers from gui to vui design. In CUI 2021 - 3rd Conference on Conversational User Interfaces (New York, NY, USA, 2021), CUI '21, Association for Computing Machinery.

[18] ORSO, V., VERÍ, D., MINATO, R., SPERDUTI, A., AND GAMBERINI, L. Are professional kitchens ready for dummies? a comparative usability evaluation between expert and non-expert users. In Human-Computer Interaction. Design and User Experience Case Studies (Cham, 2021), M. Kurosu, Ed., Springer International Publishing, pp. 418–428.

[19] Portigal, S. Interviewing users: how to uncover compelling insights. Rosenfeld Media, 2013.

[20] Preece, J., Rogers, Y., and Sharp, H. Interaction design: beyond human-computer interaction, 4th ed. John Wiley & Sons, 2015.

[21] Rubin, J., and Chisnell, D. Handbook of usability testing: how to plan, design and conduct effective tests, 2nd ed. John Wiley & Sons, 2008.

[22] Sauro, J. A practical guide to the system usability scale: Background, benchmarks & best practices. Measuring Usability LLC Denver, CO, 2011.

[23] Sauro, J., and Lewis, J. R. Quantifying the user experience: Practical statistics for user research. Morgan Kaufmann, 2016.

[24] SCHMIDT, A., ALT, F., AND MÄKELÄ, V. ¨ Evaluation in Human-Computer Interaction – Beyond Lab Studies. Association for Computing Machinery, New York, NY, USA, 2021.

[25] Shneiderman, B. The new ABCs of research: Achieving breakthrough collaborations. Oxford University Press, 2016.

[26] SÁCHEZ-ADAME, L. M., MENDOZA, S., URQUIZA, J., RODRÍGUEZ, J., AND MENESES-VIVEROS, A. Towards a set of heuristics for evaluating chatbots. IEEE Latin America Transactions 19, 12 (May 2021), 2037–2045.

[27] SÁNCHEZ-ADAME, L. M., URQUIZA-YLLESCAS, J. F., AND MENDOZA, S. Measuring anticipated and episodic ux of tasks in social networks. Applied Sciences 10, 22 (Nov 2020), 8199.

[28] Tidwell, J. Designing interfaces: Patterns for effective interaction design. . O'Reilly Media, Inc.", 2010.

[29] TRIBERTI, S., DI NATALE, A. F., AND GAGGIOLI, A. Flowing Technologies: The Role of Flow and Related Constructs in Human-Computer Interaction. Springer International Publishing, Cham, 2021, pp. 393–416.

[30] VASSEUR, A., LEGER, P.-M., COURTEMANCHE, F., LABONTE-LEMOYNE, E., GEORGES, V., VALIQUETTE, A., BRIEUGNE, D., RUCCO, E., COURSARIS, C., FREDETTE, M., AND SENECAL, S. Distributed remote psychophysiological data collection for ux evaluation: A 4 pilot project. In Human-Computer Interaction. Theory, Methods and Tools (Cham, 2021), M. Kurosu, Ed., Springer International Publishing, pp. 255–267.

Tópicos selectos en Computación Sustentable

Objetivo

El alumno aprenderá los conceptos, problemas, tendencias y temas involucrados en el estudio de la Computación Sustentable.

Temario

1. Introducción a la computación sustentable.
    - 1.1 Motivación de la computación sustentable
    - 1.2 Definiciones y alcances
    - 1.3 Problemas en computación sustentable
    - 1.4 Sistemas computacionales móviles y computación sustentable
    - 1.5 Inteligencia artificial y computación sustentable

2. Computación Verde.
    - 2.1 Definición
    - 2.2 Centros de datos
    - 2.3 Cluster de computadoras de alto rendimiento
    - 2.3 Indicadores
    - 2.4 Casos de estudio

3. Sustentabilidad computacional.
    - 3.1 Motivación y modelo de sustentabilidad
    - 3.2 Diseño sustentable asistido por computadora
    - 3.3 Sistemas de eSalud
    - 3.4 Diseño urbano y ciudades inteligentes
    - 3.5 Clima y energía

4. Computación sustentable para sistemas de Computación de Alto Rendimiento
    - 4.1 Clusters de HPC y problema exaflop
    - 4.2 Tecnologías para reducir el consumo de potencia y de energía
    - 4.3 Sistemas heterogeneos
    - 4.4 Despachadores

5. Ecodiseño de software
    - 5.1 Tendencias en ingeniería de software
    - 5.2 Servicios y micro servicios
    - 5.3 Virtualización y contenedores
    - 5.4 Métricas e indicadores
    - 5.5 Metodologías de software

Bibliografía

1. Sangaiah, Arun Kumar, et al., eds. Intelligent Decision Support Systems for Sustainable Computing: Paradigms and Applications. Vol. 705. Springer, 2017.

2. Law, Kris MY, et al., eds. Managing IoT and Mobile Technologies with Innovation, Trust, and

Sustainable Computing. CRC Press, 2021.

3. Pande, Partha Pratim, Amlan Ganguly, and Krishnendu Chakrabarty. Design Technologies for Green and Sustainable Computing Systems. Springer, 2013.

4. Hurson, Ali R. Green and Sustainable Computing: Part I. Academic Press, 2012.

5. Hu, Wen-Chen, ed. Sustainable ICTs and management systems for green computing. IGI Global, 2012.

6. Smith, Bud E. Green computing: Tools and techniques for saving energy, money, and resources. CRC Press, 2013.

7. Feng, Wu-chun, ed. The Green Computing Book: Tackling Energy Efficiency at Large Scale. CRC Press, 2014.

8. Pruhs, Kirk. "Green computing algorithmics." Computing and Software Science. Springer, Cham, 2019. 161-183.

9. Hooper, Andy. "Green computing." Communication of the ACM 51.10 (2008): 11-13.

10. Sharma, Rohit, et al., eds. Big Data Analysis for Green Computing: Concepts and Applications. CRC Press, 2021.

11. Ranka, Sanjay. Handbook of Energy-Aware and Green Computing, Volume 1. Chapman and Hall/CRC, 2012.

12. Banerjee, Sourav, Chinmay Chakraborty, and Kousik Dasgupta, eds. Green Computing and Predictive Analytics for Healthcare. CRC Press, 2020.

13. Zahran, Mohamed. Heterogeneous computing: Hardware and software perspectives. Morgan & Claypool, 2019.

14. Terzo, Olivier, et al., eds. Heterogeneous Computing Architectures: Challenges and Vision. CRC Press, 2019

Tópicos Selectos de Sistemas Ciber-físicos

Objetivo

En este curso tiene como objetivo que los alumnos comprendan y apliquen tecnologías y conceptos relacionados con arquitecturas de software para el desarrollo de sistemas ciber-físicos.

Temario

1. Fundamentos de los sistemas ciber-físicos (SCF)

   a. Introducción a los SCF
   b. Requerimientos para el desarrollo de SCF
   c. Aplicaciones de los SCF
   d. Evolución de los SCF
   g. Retos y oportunidades

2. Sistemas ciber-físicos aplicados en diferentes ámbitos

   a. Sistemas ciber-físicos médicos
   b. Sistemas ciber-físicos en el transporte
   c. Sistemas ciber-físicos en la industria 4.0

3. Arquitecturas de software

   a. Estándares IEEE 1471-2000 y ISO/IEC/IEEE 42010
   b. Arquitectura orientada a servicios
   c. Bus de mensajes
   d. N-tier
   e. Arquitectura SaaS multi-tenant

4. Arquitecturas de software basadas en computo Cloud, Fog y Edge

   a. Arquitecturas basadas en computo en la Nube
   b. Arquitecturas basadas en computo Fog y Edge
   c. Arquitecturas basadas en computo Dew

5. Sistemas ciber-físicos médicos (SCFM): caso de estudio

   a. Importancia de los SFC en el cuidado de la salud
   b. Requerimientos arquitecturales para los SCFM
   c. Arquitecturas de software propuesta para la implementación de SCFM
   d. Escenarios de evaluación y aspectos evaluados de las arquitecturas de software

Bibliografía

1. Cyber-Physical Systems in the Built Environment, C. J. Anumba, N. Roofigari-Esfahan, Editorial: Springer, 219, First Edition. ISBN: 978-3-030-41559-4
2. Cyber-Physical Systems: Architecture, Security and Application, S. Guo, D. Zeng, Editorial: Springer, 2019, First Edition, ISBN: 978-3-319-92563-9
3. Internet Computing: Principles of Distributed Systems and Emerging Internet-Based Technologies, A. Sunyaev, Editorial: Springer Cham, 2020, First Edition. ISBN: 978-3-030-34956-1
4. Fog and Edge Computing: Principles and Paradigms, R. Buyya, S. Narayana Srirama, Editorial: Wiley ,2019, First Edition, ISBN: 9781119524984

Tópicos Selectos de Sistemas Distribuidos: Computación en Internet

Objetivo

En este curso tiene como objetivo que los alumnos comprendan y apliquen tecnologías y conceptos relacionados con sistemas distribuidos a través del campo científico de la computación en Internet.

Temario

1. Introducción a la computación en Internet

    a. Evolución de la computación en Internet
    b. Definición de computación en Internet
    c. Sistemas de información distribuidos para computación en Internet
    d. Ejemplos de aplicaciones de computación en Internet

2. Arquitecturas de sistemas de información

    a. Definición de arquitecturas de sistemas de información
    b. Principios de arquitecturas de sistemas de información
    c. Vistas arquitecturales
    d. Patrones arquitecturales

3. Diseño de arquitecturas de sistemas de información

    a. Diseño arquitectural
    b. Calidad de las arquitecturas de sistemas de información
    c. Proceso de diseño de las arquitecturas de sistemas de información
    d. Arquitecturas de Internet

4. Middleware y Servicios Web (Web services)

    a. Introducción al Middleware
    b. Llamadas a procedimientos remotos
    c. Categorías de Middlewares
    d. Introducción a servicios web
    e. Tecnologías Web esenciales
    f. Arquitecturas de servicios Web

5. Computo en la nube (Cloud computing)

    a. Introducción al CC
    b. Aspectos esenciales de la prestación de servicios en nube
    c. Oportunidades y retos del CC
    d. Seguridad y protección de datos en entornos de nube

6. Fog and Edge computing (FC y EC)

    a. Fundamentos del FC y EC
    b. Oportunidades y retos del FC y EC

c. FC y EC en la práctica

Bibliografía

1. Internet Computing: Principles of Distributed Systems and Emerging Internet-Based Technologies, A. Sunyaev, Editorial: Springer Cham, 2020, First Edition. ISBN: 978-3-030-34956-1

2. Distributed Systems: Concepts and Design. G. Coulouris, J. Dollimore, T. Kindberg, Editorial: Pearson, 2011, Fifth edition. ISBN: 978-0132143011

3. Computer Networking: A Top-Down Approach. J. F. Kurose, K. W Ross, Editorial: Pearson India Education Services Private Limited, 2022, First Edition, ISBN: 978-9356061316.

4. Fog and Edge Computing: Principles and Paradigms, R. Buyya, S. Narayana Srirama, Editorial: Wiley ,2019, First Edition, ISBN: 9781119524984