**Pontifícia Universidade Católica do Paraná**

Plano de Ensino

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Escola/ Câmpus:** | Escola Politécnica - Curitiba | | | | | |
| **Curso:** | Bacharelado em Ciência da Computação (BCC) | | | **Ano/Semestre**: | 2022-2 | |
| **Código/Nome da disciplina:** | Programação Distribuída, Paralela e Concorrente (PDPC) | | | | | |
| **Carga Horária:** | 120 horas-aula | | | | | |
| **Requisitos:** | Não há | | | | | |
| **Créditos:** |  | **Período:** 4 | **Turma**: U | | | **Turno:** Manhã e Noite |
| **Professor Responsável:** | Alcides Calsavara | | | | | |

1. Ementa:

A disciplina, de natureza teórico-prática, é ofertada no quarto período do Bacharelado em Ciência da Computação. O estudante aplica algoritmos e técnicas de programação distribuída, processamento paralelo e controle de concorrência na construção de sistemas computacionais complexos. Ao término da disciplina, é capaz de codificar aplicações que exploram eficientemente os recursos disponíveis no ambiente computacional para processamento, armazenamento de dados e comunicação entre processos, propiciando ganhos no seu desempenho e na sua robustez. Para frequentar a disciplina, é desejável conhecimento em algoritmos, estruturas de dados e programação orientada a objetos.

2. Relação com disciplinas precedentes e posteriores

Esta disciplina é parte integrante do eixo de programação do curso, fornecendo a base conceitual e tecnológica necessária para a programação de sistemas computacionais complexos e de grande porte. Não possui pré-requisitos, mas emprega os resultados de aprendizagem desenvolvidos em Raciocínio Algorítmico (1o. Período), Programação Orientada a Objetos (2o. Período) e Resolução de Problemas Estruturados em Computação (3o. Período). Ainda, contribui com resultados de aprendizagem para as seguintes disciplinas:

1. Big Data (5o. Período): no processamento paralelo de grandes massas de dados
2. Conectividade e Sistemas Ciberfísicos (5o. Período): na programação de aplicações distribuídas
3. Aprendizagem de Máquina (5o. Período): no processamento paralelo de algoritmos de alta demanda de processamento
4. Arquitetura de Software (5o. Período): na domínio de requisitos não funcionais em sistemas computacionais complexos
5. Data Science (6o. Período): no processamento paralelo de grandes massas de dados e de algoritmos de alta demanda de processamento
6. Arquitetura de Sistemas Distribuídos, Paralelos e Concorrentes (7o. Período): na verificação experimental de soluções alternativas para a construção de sistemas complexos
7. Complexidade de Algoritmos (7o. Período): na validação experimental de resultados obtidos analiticamente
8. Avaliação de Desempenho de Sistemas (8o. Período): na validação experimental de resultados obtidos analiticamente

3. Temas de estudo

1. Processos e threads
2. Comunicação entre threads por meio de memória compartilhada
3. Sincronização entre threads baseada em semáforos
4. Padrões de sincronização baseados em semáforos: sinalização, mutex, rendezvous, barreira simples, barreira reusável, catraca
5. Propriedades de correção de aplicações concorrentes: deadlock, liveness, starvation, fairness
6. Modelos de paralelismo em algoritmos: iterativo e recursivo
7. Avaliação de desempenho de aplicações paralelas: eficiência, speedup, escalabilidade, Lei de Amdahl
8. Técnicas de paralelização de aplicações: análise de dependências, detecção de hotspots
9. Testes de correção de aplicações paralelas e concorrentes
10. Servidores multithreaded
11. Controle de concorrência baseado em monitores
12. Protocolos de transporte de mensagens entre processos distribuídos
13. Transmissão de dados estruturados entre processos distribuídos por meio de mensagens

4. Resultados de Aprendizagem

|  |  |  |
| --- | --- | --- |
| Resultados de Aprendizagem | Temas de Estudo | Elemento de Competência (Internaliza, Mobiliza, Certifica) e Competência |
| RA1. Aplicar adequadamente os paradigmas de desenvolvimento de aplicações paralelas, distribuídas e concorrentes de acordo com os requisitos da aplicação e das condições do ambiente computacional | * Processos e threads * Servidores multithreaded * Comunicação entre threads por meio de memória compartilhada * Sincronização entre threads baseada em semáforos * Controle de concorrência baseado em monitores * Protocolos de transporte de mensagens entre processos distribuídos | EC3. Selecionar o paradigma de programação adequado para problemas complexos e não estruturados (I) |
| RA2. Aplicar adequadamente as técnicas padrão de design e implementação de aplicações paralelas, distribuídas e concorrentes de acordo com os requisitos da aplicação e das condições do ambiente computacional | * Padrões de sincronização baseados em semáforos: sinalização, mutex, rendezvous, barreira simples, barreira reusável, catraca * Modelos de paralelismo em algoritmos: iterativo e recursivo * Técnicas de paralelização de aplicações: análise de dependências, detecção de hotspots * Transmissão de dados estruturados entre processos distribuídos por meio de mensagens | EC4. Aplicar padrões adequados na solução e codificação de problemas complexos e não estruturados (I) |
| RA3. Construir aplicações paralelas, distribuídas e concorrentes que apresentem funcionamento correto e eficiente, usando ambiente de programação adequado | * Propriedades de correção de aplicações concorrentes: deadlock, liveness, starvation, fairness * Testes de correção de aplicações paralelas e concorrentes * Avaliação de desempenho de aplicações paralelas: eficiência, speedup, escalabilidade, Lei de Amdahl | EC8. Selecionar ambientes de desenvolvimento apropriados para sistemas computacionais (I) |

5. Mapa Mental



6. Metodologia e Avaliação

|  |  |  |  |
| --- | --- | --- | --- |
| Alinhamento Construtivo | | | |
| Resultado de aprendizagem | Indicadores de desempenho | Processos de Avaliação | Métodos ou técnicas empregados\*\* |
| **RA1:** Aplicar adequadamente os paradigmas de desenvolvimento de aplicações paralelas, distribuídas e concorrentes de acordo com os requisitos da aplicação e das condições do ambiente computacional | **ID1.1:** Implementa comunicação entre threads por meio de variável compartilhada  **ID1.2:** Implementa sincronização entre threads usando semáforos  **ID1.3**: Implementa controle de concorrência usando monitores  **ID1.4:** Implementa corretamente servidores multithreaded  **ID1.5:** Implementa comunicação entre processos distribuídos usando protocolos de transporte de mensagens | 1. PBL em equipe, com avaliação formativa e feedback na semana seguinte 2. PjBL individual, com avaliação somativa e feedback no momento da defesa | 1. Aula expositiva dialogada 2. PBL 3. PjBL 4. Portfólio (Lista de Exercícios)   Será utilizado o ambiente virtual de aprendizagem para organização do material didático e para a interação com os estudantes. |
| **RA2:** Aplicar adequadamente as técnicas padrão de design e implementação de aplicações paralelas, distribuídas e concorrentes de acordo com os requisitos da aplicação e das condições do ambiente computacional | **ID2.1:** Emprega corretamente os padrões de sincronização baseados em semáforos  **ID2.2:** Emprega corretamente o modelo de paralelismo iterativo  **ID2.3:** Emprega corretamente o modelo de paralelismo recursivo  **ID2.4:** Implementa atransmissão de dados estruturados entre processos distribuídos por meio de mensagens | 1. PBL em equipe, com avaliação formativa e feedback na semana seguinte 2. PjBL individual, com avaliação somativa e feedback no momento da defesa | 1. Aula expositiva dialogada 2. PBL 3. PjBL 4. Portfólio (Lista de Exercícios)   Será utilizado o ambiente virtual de aprendizagem para organização do material didático e para a interação com os estudantes. |
| **RA3:** Construir aplicações paralelas, distribuídas e concorrentes que apresentem funcionamento correto e eficiente, usando ambiente de programação adequado | **ID3.1:** Analisa a correção do código de uma aplicação concorrente  **ID3.2**: Valida aplicações concorrentes e paralelas por meio de testes apropriados  **ID3.3:** Avalia o desempenho de aplicações paralelas quanto a eficiência, speedup e escalabilidade  **ID3.4:** Analisa corretamente as dependências e identifica os hotspots em aplicações paralelas | 1. PBL em equipe, com avaliação formativa e feedback na semana seguinte 2. PjBL individual, com avaliação somativa e feedback no momento da defesa | 1. Aula expositiva dialogada 2. PBL 3. PjBL   Será utilizado o ambiente virtual de aprendizagem para organização do material didático e para a interação com os estudantes. |

A aprovação do estudante na disciplina está condicionada à obtenção de uma nota final 7,0 (sete), no mínimo. A nota de cada estudante é calculada como uma média ponderada, de acordo com os pesos definidos na tabela a seguir.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipo de trabalho** | **Descrição** | **Forma de realização** | **Peso na nota do semestre** |
| PBL | Trabalhos semanais (aproximadamente, 10 entregas, todas com o mesmo peso na nota) | Equipes de, no máximo, 4 estudantes | 60% |
| PjBL | Trabalho de longa duração (aproximadamente, um mês) e de tema livre | Individual | 40% |

**Critério de aprovação**: nota do semestre igual ou superior a 7 (sete) e

frequência mínima de 75%.

O estudante pode recuperar a nota de todo e qualquer trabalho:

1. Se um PBL não for entregue no prazo, o estudante poderá entregar com atraso de até duas semanas após o prazo, sendo a nota máxima igual a 7 (sete).
2. Se um PBL for entregue no prazo e a nota atribuída seja inferior a 7 (sete), o estudante poderá refazer a entrega até uma semana após a publicação da nota.
3. Se o PjBL não for entregue no prazo ou caso tenha sido entregue e a nota atribuída seja inferior a 7 (sete), o estudante poderá entregar até a “Semana Estendida de Recuperação”, conforme Calendário Acadêmico da PUCPR, sendo a nota máxima igual a 7 (sete).

7. Cronograma de atividades

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Período**  **(horas aula totais,**  **(dia, semana, quinzena, mês)** | **RAs** | **Atividades pedagógicas** | **Em aula / TDE** | **Carga horária da atividade** |
| Semana 1 | 1 | Thread. Semáforo. Variável compartilhada. Padrão Mutex. PBL 01: Exercício Mutex. | Em aula | 6 |
| Semana 2 | 2 | Padrão Sinalização e Padrão Scoreboard: Problema do Jantar dos Filósofos, Problema do Jantar dos Selvagens. PBL 02: Padrão Sinalização. | Em aula | 6 |
| Semana 3 | 2 | Padrão Produtor-Consumidor. PBL 03: Exercício Produtor-Consumidor. | Em aula | 6 |
| Semana 4 | 2 | PBL 03: Exercício Produtor-Consumidor. | Em aula | 6 |
| Semana 5 | 2 | Padrão Rendezvous. Padrão Barreira Simples. Padrão Barreira Dupla. PBL 04: Exercício Barreira Dupla. | Em aula | 6 |
|  | 1 e 2 | Lista de exercícios sobre semáforos | TDE | 4 |
| Semana 6 | 2 | PBL 04: Exercício Barreira Dupla. | Em aula | 6 |
| Semana 7 | 2 | Monitor: Problema do Jantar dos Selvagens. Problema da Barbearia. PBL 05: Exercício Monitor | Em aula | 6 |
| Semana 8 | 2 | PBL 05: Exercício Monitor | Em aula | 6 |
|  | 2 | Lista de exercícios sobre monitores | TDE | 4 ou 6 |
| Semana 9 | 2 e 3 | Modelo de paralelismo iterativo. Avaliação de desempenho de aplicações paralelas. PBL 06: Paralelismo Iterativo. | Em aula | 6 |
| Semana 10 | 2 e 3 | Modelo de paralelismo recursivo. PBL 07: Paralelismo Recursivo. | Em aula | 6 |
| Semana 11 | 2 e 3 | PBL 07: Paralelismo Recursivo. | Em aula | 6 |
|  | 2 e 3 | Lista de exercícios sobre paralelismo | TDE | 4 ou 6 |
| Semana 12 | 1 e 2 | Comunicação entre processos distribuídos. PBL 08: Programação Distribuída. | Em aula | 6 |
| Semana 13 | 1 e 2 | PBL 08: Programação Distribuída | Em aula | 6 |
| Semana 14 | 1, 2 e 3 | PjBL | Em aula | 6 |
| Semana 15 | 1, 2 e 3 | PjBL | Em aula | 6 |
| Semana 16 | 1, 2 e 3 | PjBL | Em aula | 6 |

8. Bibliografia

Todas as atividades contarão com o apoio de material preparado pelo professor sob a forma de slides, vídeos e exemplos de código disponibilizados no ambiente virtual de aprendizagem. Os seguintes livros poderão ser consultados na biblioteca da PUCPR pelos estudantes como fonte complementar:

1. TANENBAUM, Andrew S.; STEEN, Maarten van. Sistemas distribuídos: princípios e paradigmas. 2. ed.; Ed. Prentice-Hall, 2007.
2. ANDREWS, Gregory R. **Foundations of multithreaded, parallel, and distributed programming.**Massachusetts: Addison-Wesley, 2000. xx, 664 p. ISBN 0-201-35752-6.
3. SPRINGERLINK (ONLINE SERVICE) **Distributed and Parallel Systems:**From Instruction Parallelism to Cluster Computing. XII, 233 p (The Springer International Series in Engineering and Computer Science, 0893-3405 ; 567). ISBN 9781461544890.
4. LEA, Douglas. **Concurrent programming in Java:**design principles and patterns. 2nd ed. Reading: Addison-Wesley, c2000. ix, 411 p. (Java series). ISBN 0-201-31009-0.
5. BEN-ARI, M. **Principles of concurrent and distributed programming.**2nd ed. Harlow, England: Addison-Wesley, 2006. xv, 361 p ISBN 0-321-31283-X (broch.).
6. COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim. **Sistemas distribuídos: conceitos e projeto**. Ed. Bookman, 2007

8.2. Complementares

1. Andrews, Gregory R.; Olsson, Ronald A. "The SR programming language**:** concurrency in practice". Redwood City: Benjamin/Cummings, c1993. 344 p. ISBN 0-8053-0088-0
2. Butenhof, D. R. (2006) “Programming with POSIX Threads”, Addison-Wesley, 2006.
3. Chapman, B.; Jost, G.; Van der Pas, R. (2008) “Using OpenMP – Portable shared memory parallel programming”. The MIT Press, 2008.
4. Downey, A. B. "The Little Book of Semaphores", Second Edition, 2016.
5. Kirk, D. B., Hwu, W.W. (2010) “Programming massively parallel processors: a hands-on approach”. Morgan Kaufman, 2010.
6. Mattson, T.G., Sanders, B., Massingill, B. “Patterns for Parallel Programming”, Addison-Wesley Professional, 2004.
7. McCool, M., Reinders, J. and Robinson, A. (2012) "Structured Parallel Programming: Patterns for Efficient Computation", Morgan Kaufmann, 1st edition.
8. Pacheco, P.S. “An Introduction to Parallel Programming”, Morgan Kaufmann, 2011.
9. Van der Pas, Ruud; Stotzer, Eric; Terboven, Christian. (2017) "Using OpenMP - The Next Step: Affinity, Accelerators, Tasking, and SIMD". The MIT press, Scientific and Engineering Computation series. Outubro, 2017
10. Lynch, Nancy A. "Distributed algorithms". San Francisco: M. Kaufmann 1997.
11. Brian Goetz, Tim Peierls, Joshua Bloch, Joseph Bowbeer, David Holmes, Doug Lea. "Java Concurrency in Practice". Addison-Wesley Professional; 1 edition (May 19, 2006).
12. Giancarlo Zaccone. "Python Parallel Programming Cookbook". Packt Publishing - ebooks Account (September 1, 2015).
13. Maurice Herlihy, Nir Shavit. "The Art of Multiprocessor Programming". Morgan Kaufmann; 1 edition (June 5, 2012).
14. Thomas Rauber (Author), Gudula Rünger. "Parallel Programming for Multicore and Cluster Systems". Springer; 2nd ed. 2013 edition (June 8, 2013).
15. Michel Raynal. "Concurrent Programming: Algorithms, Principles, and Foundations". Springer; 2013 edition (December 26, 2012).
16. Nathan Marz, James Warren. "Big Data: Principles and best practices of scalable real time data systems". Manning Publications; 1 edition (May 10, 2015).

9. Acessibilidade\*\*

Não houve necessidade de adaptação.

*\*\* conforme nota técnica conjunta número 17/2020 CGLNRS/DPR/SERES/SERES*