Sistemas de Tempo Real:

O Tempo Real

Rômulo Silva de Oliveira
Departamento de Automação e Sistemas – DAS – UFSC

romulo@das.ufsc.br
http://www.das.ufsc.br/~romulo

Maio/2007

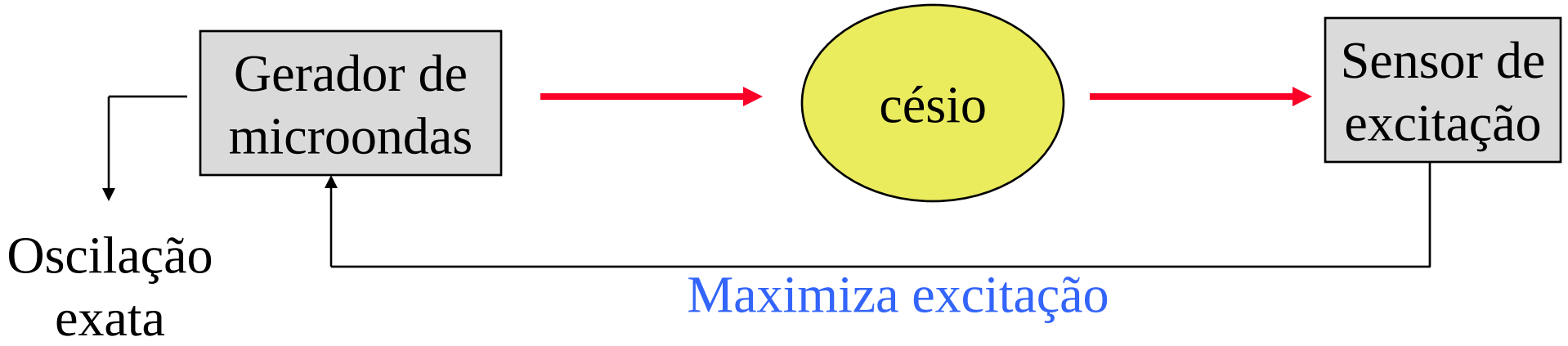
Relógio Físico - Astronômico

- Medição do tempo em termos **astronômicos**
  - Sol no ponto aparente mais alto: Trânsito Solar
  - Intervalo entre dois trânsitos solares: Dia Solar
  - 1/(24*3600) do Dia Solar: Segundo Solar

- Rotação da terra diminui com o tempo
  - 400 dias por ano a 300 milhões de anos atrás
  - Os dias estão ficando mais longos
  - Segundo Solar Médio - média de vários dias

Relógio Físico - Atômico

- Medição do tempo em termos **atômicos**
- Em 1948 foi criado o relógio atômico
  - Definição de segundo: Tempo necessário para ocorrerem 9.192.631.770 transições do átomo de césio 133
  - Por quê 9.192.631.770 ?
    - Resulta em 1 segundo solar médio no ano em que o relógio atômico foi introduzido



Relógio Físico - Atômico

- Atualmente dezenas de laboratórios no mundo possuem um relógio atômico
- Coordenados pelo Bureau International de l'Heure - BIH, Paris-França
- Cada laboratório informa o BIH quantos ticks ocorreram
- BIH faz a média, produz o
  International Atomic Time - TAI
- TAI é a média de ticks contados desde 1/jan/58

Relógio Físico - UTC

- Problema:

  24*3600 segundos do TAI é
  3mS menor que o dia solar médio
  - Dia solar médio está ficando maior
  - Rotação da terra está ficando mais lenta

- Usando somente o TAI
  - Meio-dia do relógio e do sol ficariam diferentes

Relógio Físico - UTC

- Solução:
  - BIH atrasa segundos para manter o TAI em sincronia com o sol
  - Sempre que diferença entre TAI e tempo solar passa de 800 mS
  - Até hoje cerca de 30 segundos foram atrasados
- Relógio Atômico Corrigido é chamado de

  UTC - Universal Coordinated Time

Relógio Físico – Via GPS

- GPS - Global Positioning System
- Sistema de navegação baseado em satélites orbitando a terra
- Fornece 24 horas por dia, no globo inteiro:
  - Posição precisa em 3 dimensões
  - Tempo preciso, relacionado com o UTC
- Operado pela Força Aérea americana, dirigido pelo DoD
- Projetado originalmente para fins militares
- Atualmente usado largamente para fins civis:
  - Monitoração de veículos, mapeamento, navegação
- Capacidade operacional completa em julho de 1995
- 50+ empresas fornecem 275+ modelos de receptores (1997)

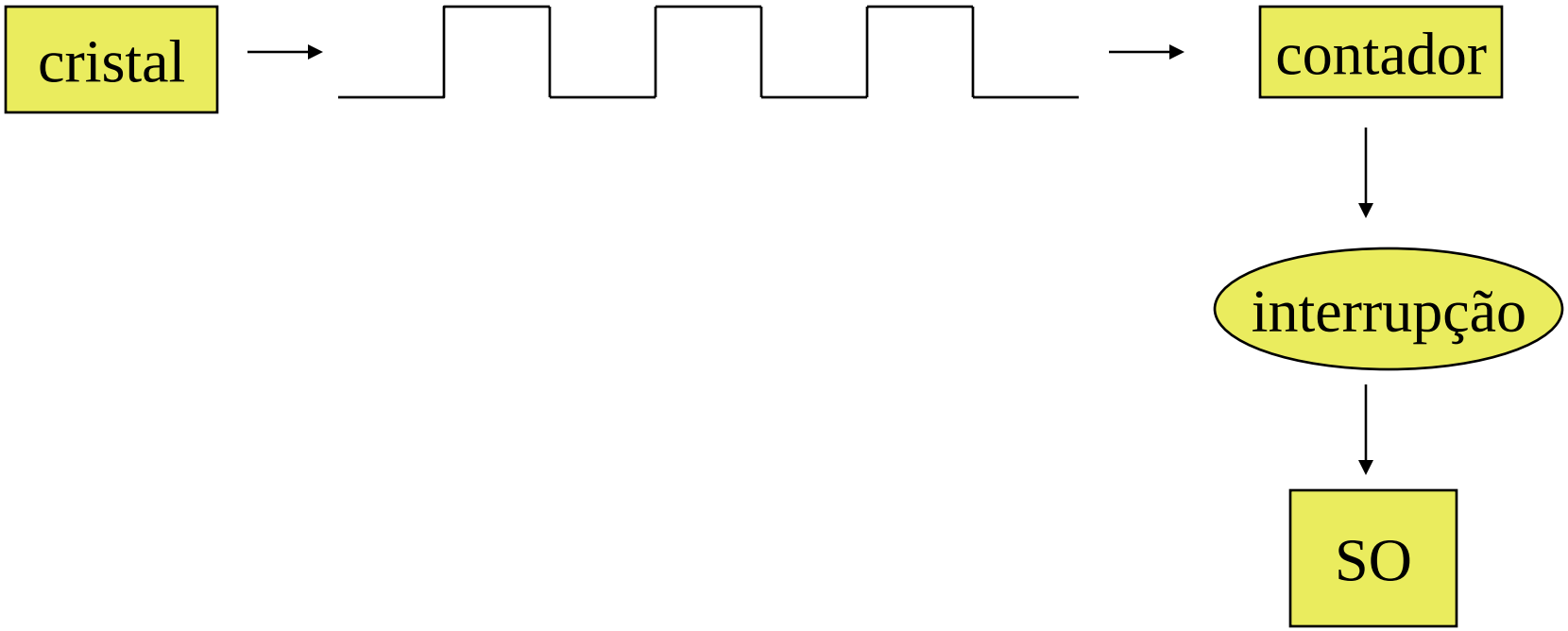
Relógio Físico – Via GPS

- Níveis de precisão diferentes para civis e militares
- GPS-Time expresso em semanas e segundos desde 6/jan/1980 UTC
- Não inclui segundos atrasados (leap-seconds), ex: 11 em 1/1/96
- Mantido por relógios atômicos em terra e nos satélites
- Receptores para tempo são diferentes dos para navegação
  - Em geral assumem posição fixa e conhecida
  - Atrasos na antena, cabo e eletrônica devem ser computados
- Preço entre US$ 200,00 e US$ 60.000,00
- Muitos sinalizam tempo via RS-232, IEEE-488 e ethernet
- Problema: Colocação da antena é crítica
  - Visada direta entre antena e satélite, reflexos, perdas no cabo
  - Determinação da posição da antena

Relógio Físico - Computadores

- Cristais de quartzo sob tensão
  - Oscilam em uma freqüência bem definida
  - Freqüência depende do tipo de cristal, lapidação e tensão
- Um contador (Timer) é iniciado com algum valor
- Cada oscilação do cristal decrementa o contador
- Quando contador chega a zero
  - Interrupção é gerada (clock tick)
  - Contador é reinicializado
- Período das interrupções definido pelo valor colocado na inicialização do timer

Relógio Físico - Computadores



- Cristais variam
  - É impossível produzir cristais com freqüência exatamente igual
  - Na prática é preciso considerar freqüências ligeiramente diferentes

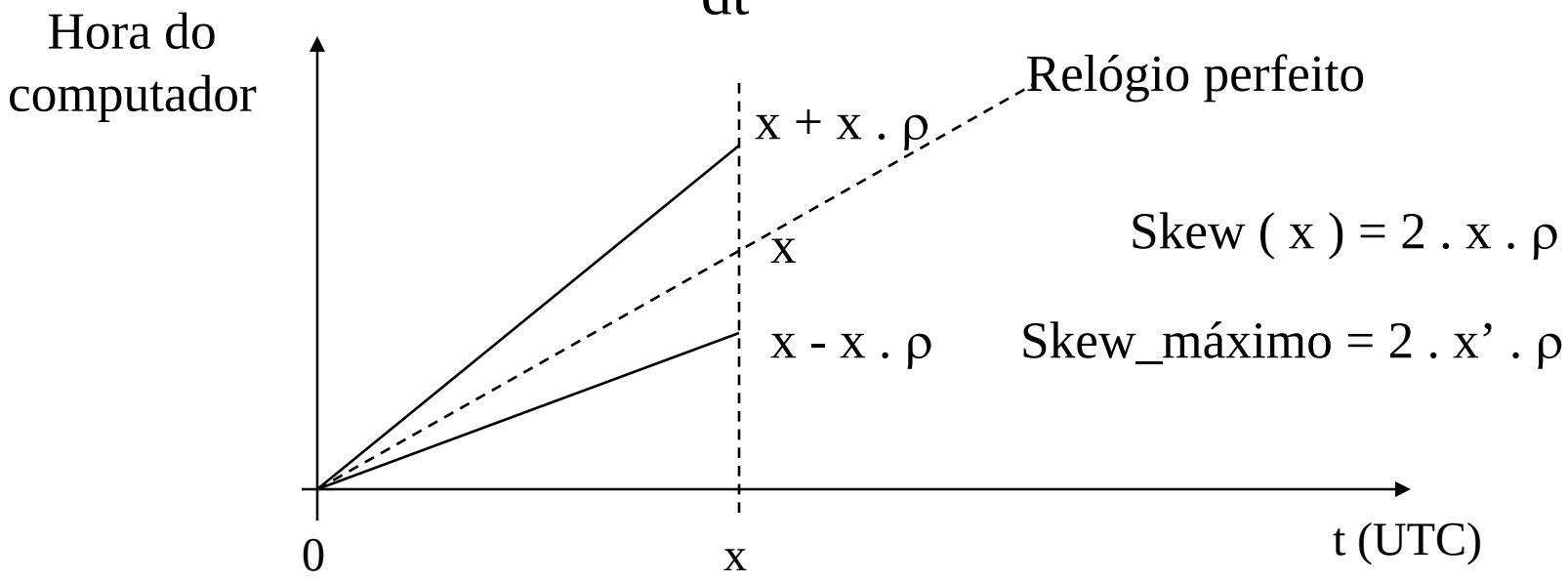
Sincronização de Relógios

- Seja **t** a hora UTC
- Seja Cp(t) a hora indicada pela máquina **p** no instante **t**
- Ideal: Para qualquer **p**, qualquer **t**, **Cp(t) = t**
- Ou ainda: **dC / dt = 1**
  - **dC / dt > 1** indica relógio muito rápido
  - **dC / dt < 1** indica relógio muito lento
- Exemplo: Timer deve gerar 60 interrupções por segundo
  - Resulta em 60 * 60 * 60 = 216000 ticks por hora
  - Erro típico é de 10^{-5}
  - Máquina típica vai gerar entre 215998 e 216002 ticks por hora
- Diferença absoluta entre relógios é chamada **SKEW**

Sincronização de Relógios

- Drift Rate máximo - ρ
  - Indica a precisão do timer
  - Definido por:

$$1-\rho \leq \frac{dC}{dt} \leq 1+\rho$$



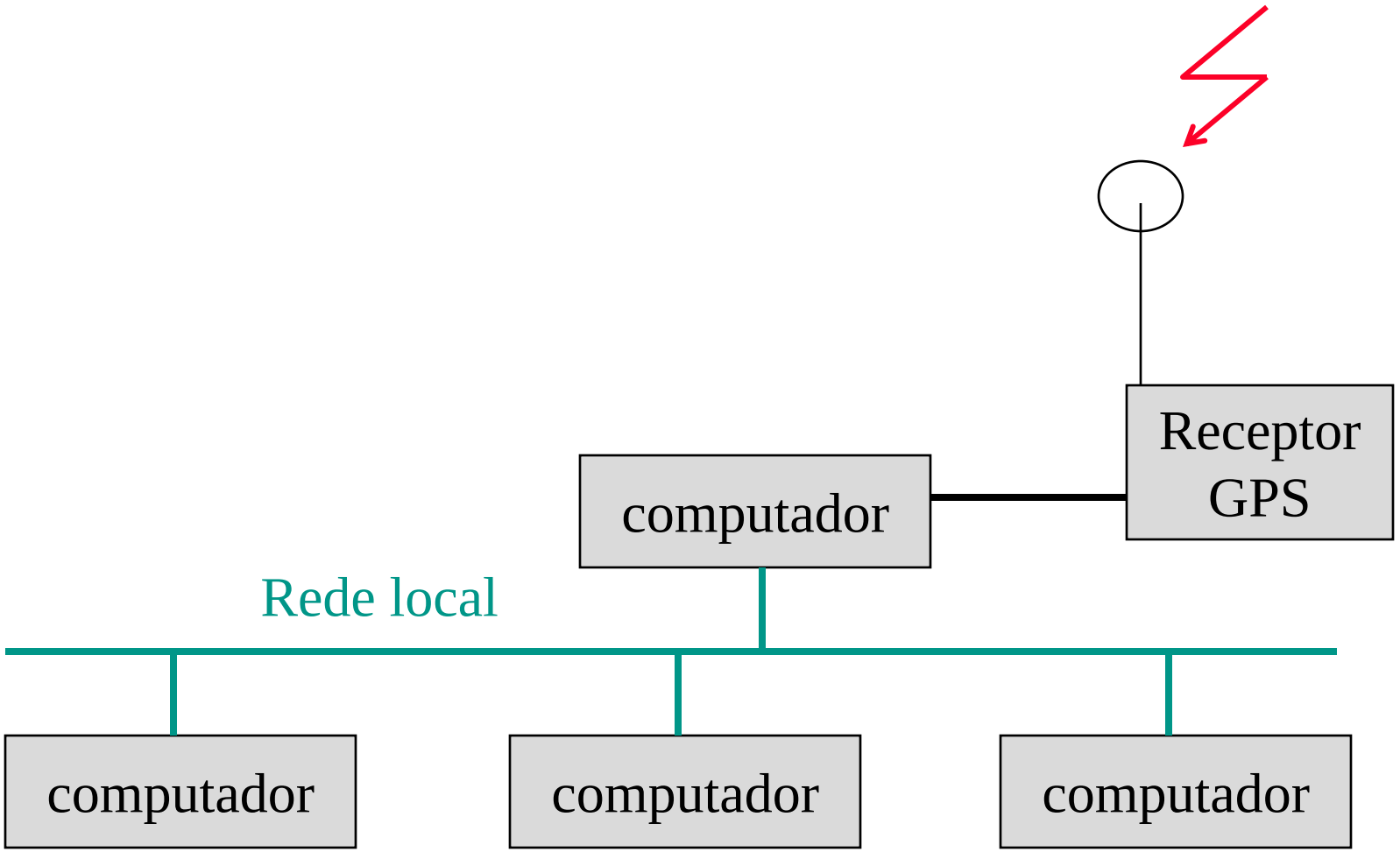
Sincronização - Solução Centralizada

- Time Server, Cristian, 1989
- Time Server tem acesso ao UTC
- Periodicamente, no máximo a cada **Skew_máximo / 2 . ρ**
  - Clientes perguntam a hora
  - Time Server responde a UTC
- Se UTC < hora local
  - Não pode atrasar o relógio
  - Faz a correção gradualmente
- Tempo para comunicação entre clientes e servidores
  - Mede atraso entre request e reply, divide por 2
  - Usa média do atraso de vários pares request-reply
  - Considera tempo de processamento do server

Sincronização - Solução Distribuída

- Não existe um servidor centralizado
- Intervalos de sincronização:
  - i-ésimo intervalo entre T0 + i.R e T0 + (i+1).R
- No início de cada intervalo:
  - Cada máquina difunde a sua hora local
  - Relógios dessincronizados, difusão não ocorre ao mesmo tempo
- Após enviar a sua hora:
  - Cada máquina coleta todas as outras difusões
  - Limita coleta a um intervalo S
  - Descarta valores extremos
  - Estima atrasos de comunicação e processamento
  - Computa nova hora local fazendo a média

Sincronização Através de GPS



Resumo

- Algumas aplicações necessitam manipular o tempo
- Relógio Físico
  - Astronômico
  - Atômico
  - UTC
- Computadores utilizam a oscilação dos cristais
- Sincronização de Relógios é por vezes necessária
  - Algoritmo centralizado
  - Algoritmo distribuído
  - Através de GPS

Sistemas de Tempo Real:

Introdução

Rômulo Silva de Oliveira
Departamento de Automação e Sistemas – DAS – UFSC

romulo@das.ufsc.br
http://www.das.ufsc.br/~romulo

Maio/2007

Caracterização

- Sistemas computacionais de tempo real:
  - Submetidos a requisitos de natureza temporal
  - Resultados devem estar corretos lógica e temporalmente
  - Requisitos definidos pelo ambiente físico
- Aspectos temporais
  - NÃO estão limitados a uma questão de maior ou menor desempenho
  - Estão diretamente associados com a funcionalidade
- Sistemas em geral:
  - "Fazer o trabalho usando o tempo necessário"
- Sistemas de tempo real:
  - "**Fazer o trabalho usando o tempo disponível**"

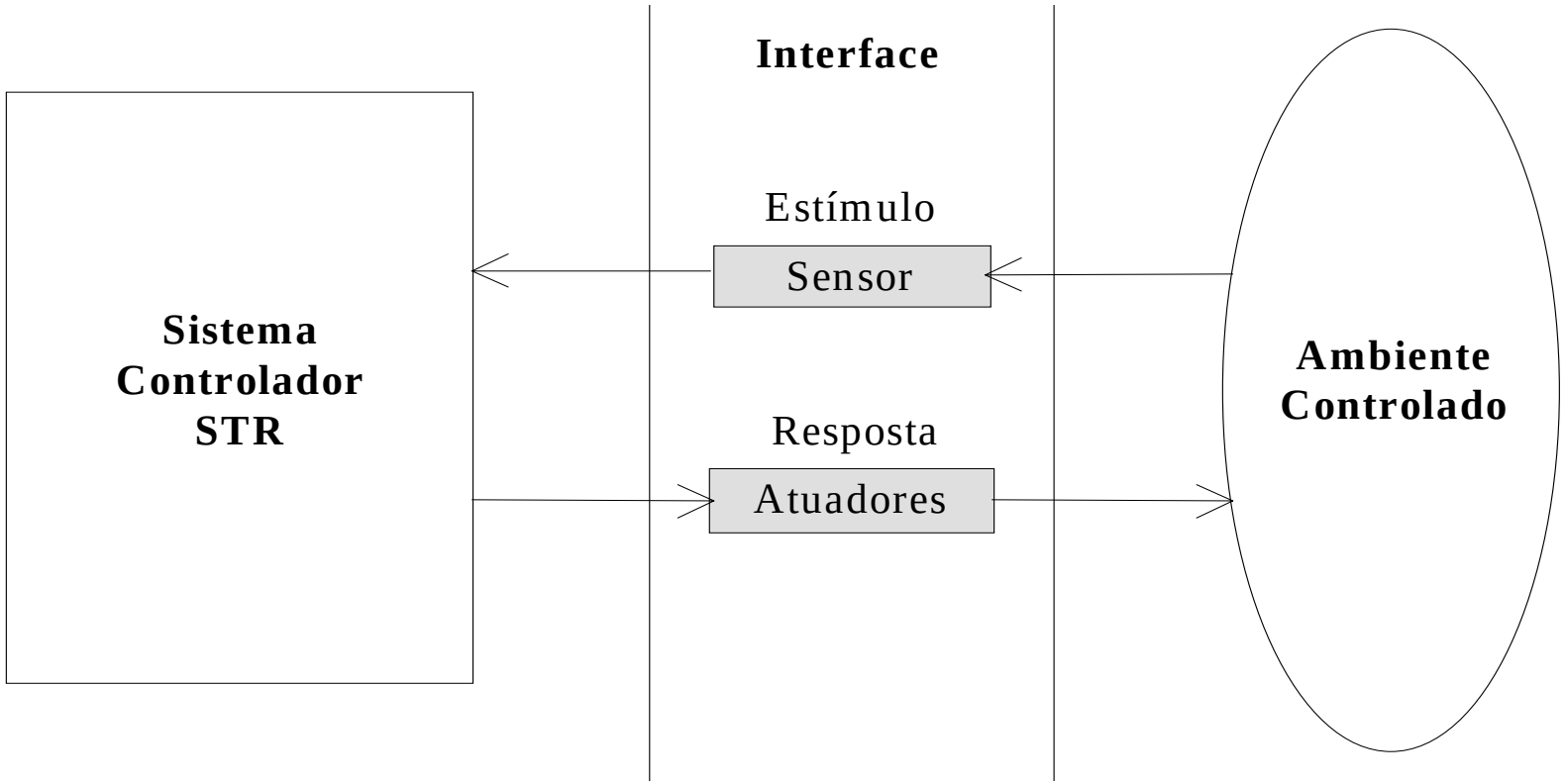
Caracterização

- Forte acoplamento de um STR com o seu Ambiente:
  - Processamento ativado por estímulos do ambiente
  - **Tempos de Resposta** delimitam **Estímulos/Respostas**
    - Processamentos devem terminar dentro de prazos (**deadlines**)
    - Se terminar fora de prazo sistema falha (**falha temporal**)

- Dados com prazos de validade:
  - Dados desatualizados ou não válidos (*outdated data*) podem conduzir a resultados ou respostas incorretas

- Fluxos de controle na execução são definidos pelo ambiente:
  - Impossibilidade em muitas aplicações do STR exercer um controle ou limitação nos estímulos provenientes do ambiente

Caracterização



Mercados

- Telecomunicações
  - Centrais telefônicas, videoconferência, groupware
- Aeroespacial
  - Automação em aeronaves, sondas espaciais
- Defesa
  - Radar, sonar, sistema guia em mísseis
- Indústria
  - Controle de processos, robôs, aquisição de dados
- Financeiro
  - Transações em bolsa, negociação automática
- Entretenimento
  - Vídeo games, vídeo sob demanda, áudio

Concepções Erradas

- Tempo real significa execução rápida
- Computadores mais rápidos vão resolver todos os problemas
- Sistemas de tempo real são:
  - pequenos
  - escritos em linguagem de montagem (assembly)
  - formados apenas por tratadores de interrupção
  - device-drivers

Conceitos Básicos

- **Tarefa** (task)
  - Segmento de código cuja execução possui atributo temporal próprio (período, deadline, etc)
  - Exemplo: método em OO, sub-rotina, trecho de um programa

- **Deadline**
  - Instante máximo desejado para a conclusão de uma tarefa

- **Deadline relativo**
  - Em relação ao início da tarefa
- **Deadline absoluto**
  - Em relação a UTC (relógio da parede)

Conceitos Básicos

- Tempo real crítico (**hard real-time**)
  - Falha temporal pode resultar em conseqüências catastróficas
  - Necessário garantir requisitos temporais em projeto
  - Exemplo: usina nuclear, industria petroquímica, mísseis

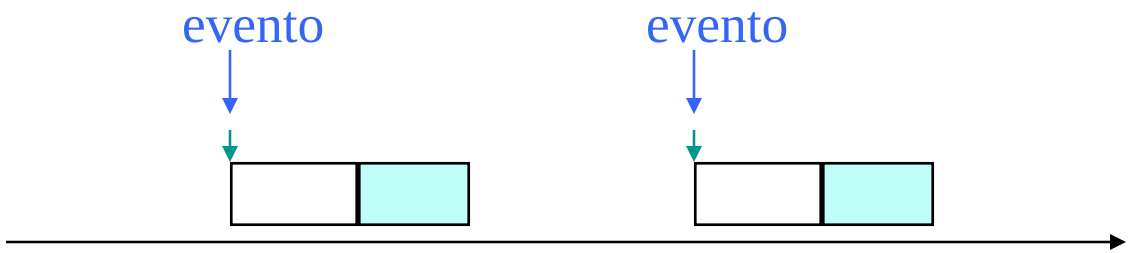
- Tempo real não crítico (**soft real-time**)
  - Requisito temporal descreve apenas comportamento desejado
  - Exemplo: multimídia

Previsibilidade

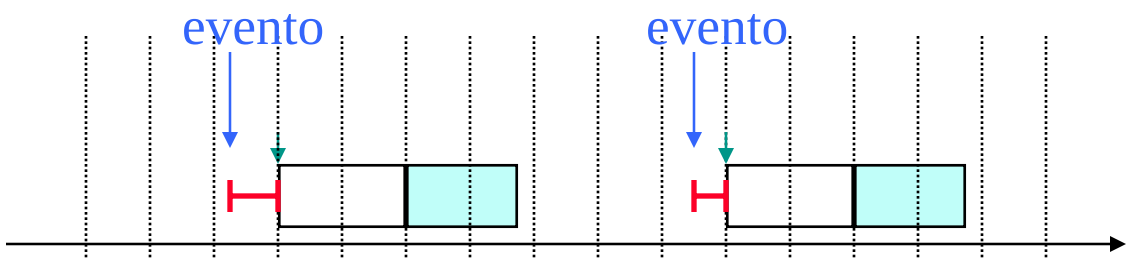
- **Previsibilidade** (“predictability”)
  - Está associada a capacidade de poder antecipar, em tempo de projeto, se os processamentos em um STR serão executados dentro de seus prazos especificados
- Associada a uma **previsão determinista**
  - todos os deadlines serão respeitados
- ou a uma **antecipação probabilista**
  - baseadas em estimativas, probabilidades são associadas a deadlines definindo as possibilidades dos mesmos serem respeitados
- A previsibilidade em STR determina implicações em todos os níveis:
  - linguagens
  - sistemas operacionais
  - comunicação
  - arquitetura do computador
  - etc

Event-Triggered x Time-Triggered

- Event-Triggered



- Time-Triggered

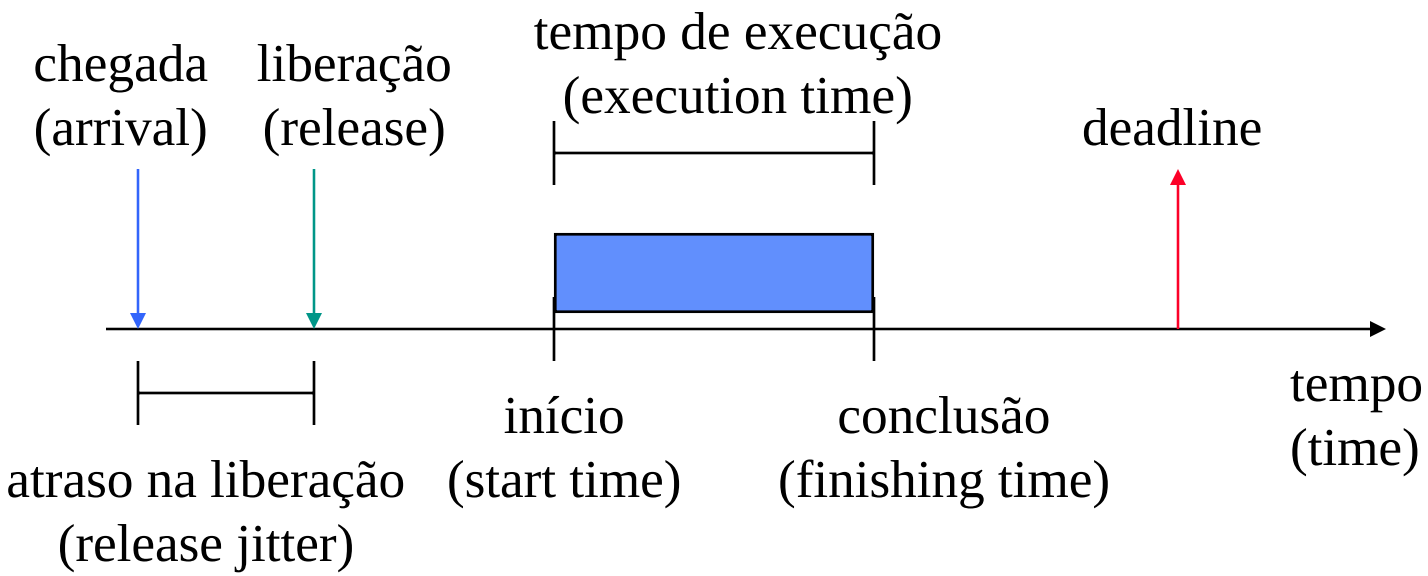


Event-Triggered x Time-Triggered

- **Dirigido por eventos** (Event-Triggered)
  - Sistema reage a eventos
  - Evento externo gera interrupção e dispara tarefa
  - Ex: Cano estourado gera chuva de eventos

- **Dirigido pelo tempo** (Time-Triggered)
  - Interrupção de relógio a cada T milisegundos (tick)
  - A cada tick alguns sensores e atuadores são acessados
  - Não existem interrupções além das do relógio
  - Problema é selecionar T (carga x atraso)
  - Ex: Relação causa-efeito em trem e cancela

Propriedades Temporais das Tarefas



Folga = Deadline - Liberação - Tempo de execução

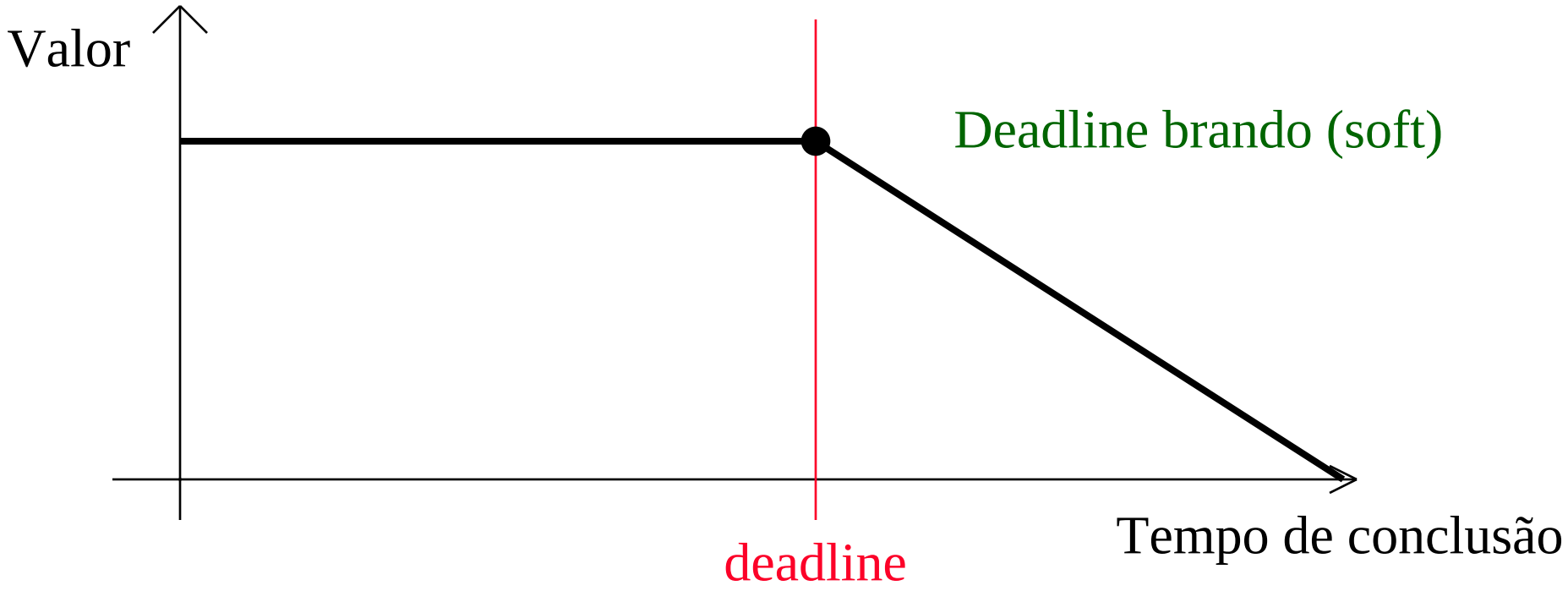
Atraso = MAX(0 , Conclusão - Deadline)

Tempo de resposta = Conclusão - Chegada

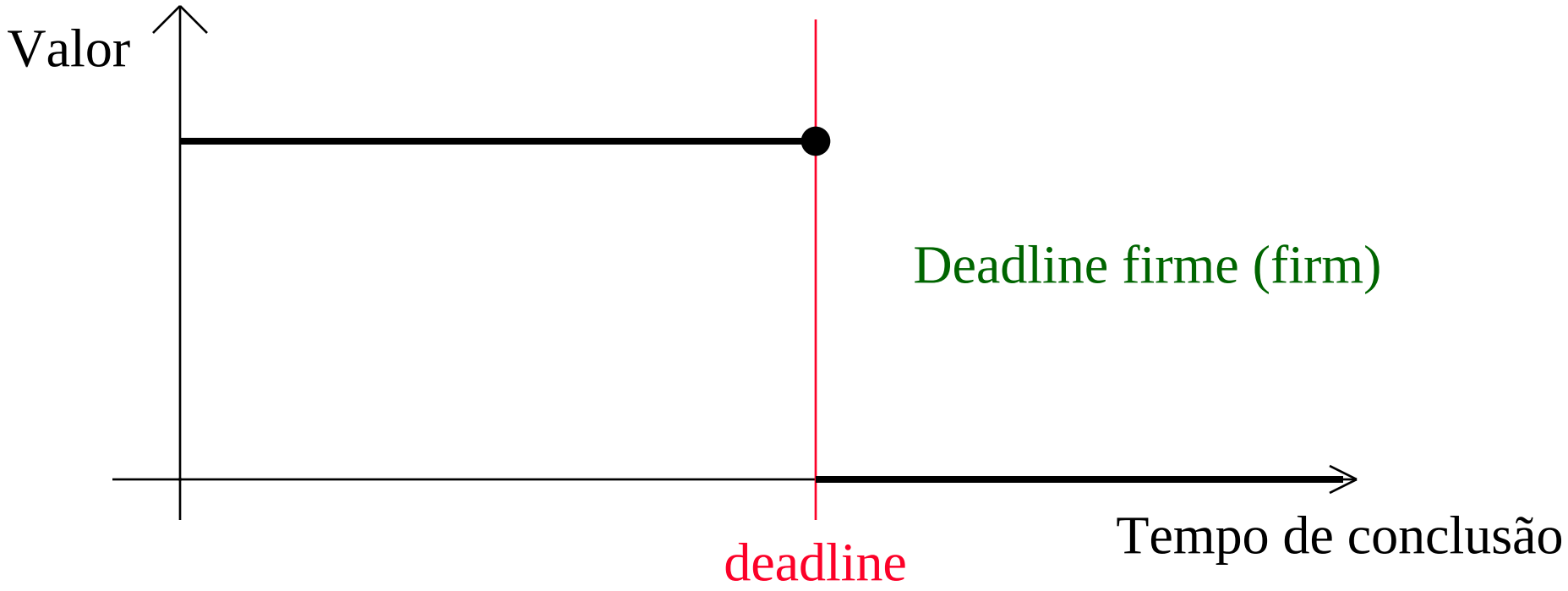
Tipos de Deadlines

- **Deadline Hard**
  - Perda do deadline pode ter consequências catastróficas
  - Exemplo: abrir válvula em duto de alta pressão
- **Deadline Firm**
  - Perda do deadline NÃO tem consequências catastróficas
  - Não existe valor em terminar a tarefa após o deadline
  - Exemplo: amostrar periodicamente valor físico
- **Deadline Soft**
  - Perda do deadline NÃO tem consequências catastróficas
  - Existe valor em terminar a tarefa com atraso
  - Exemplo: movimento de objeto em vídeo game

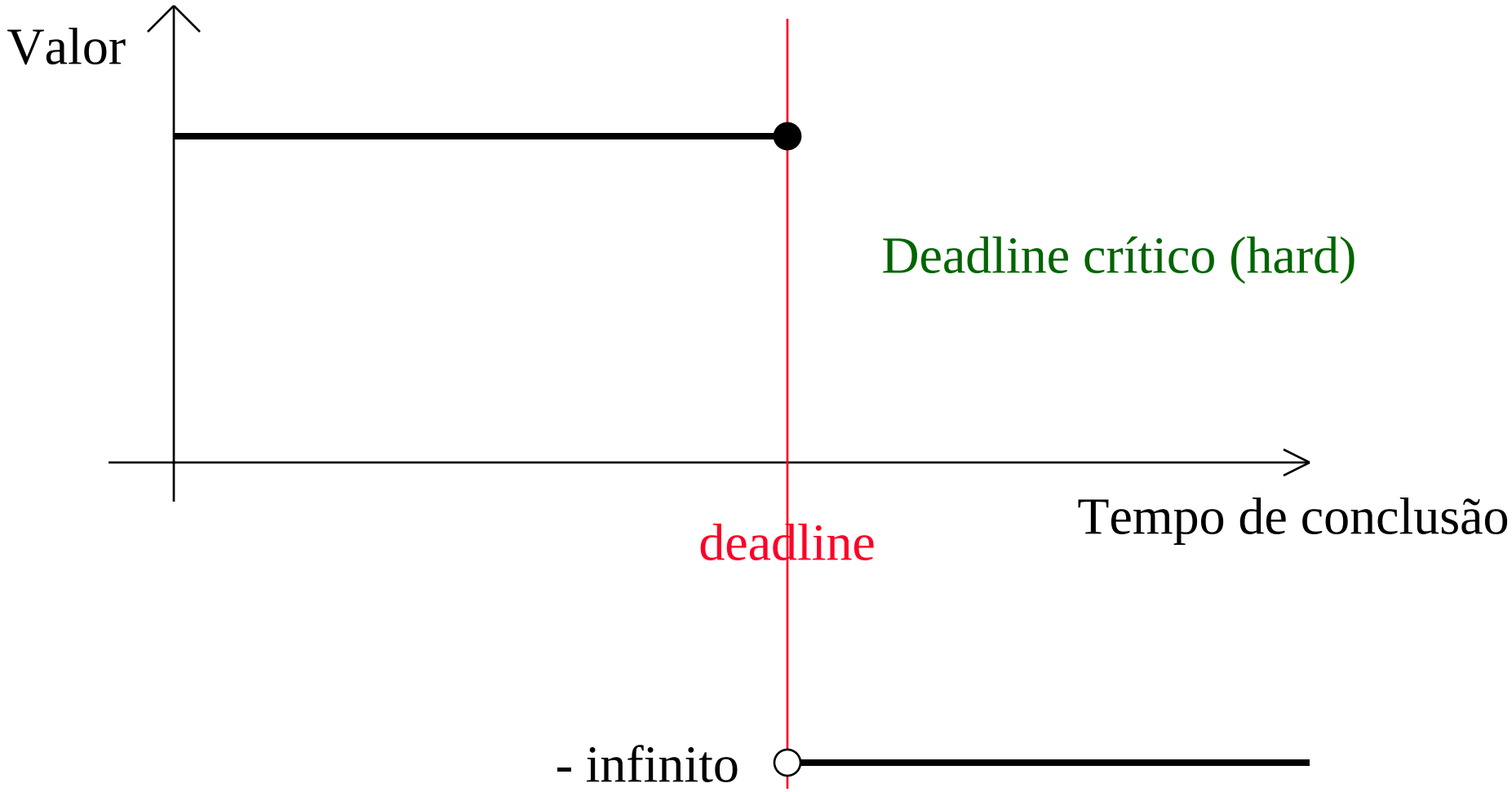
Tipos de Deadlines



Tipos de Deadlines



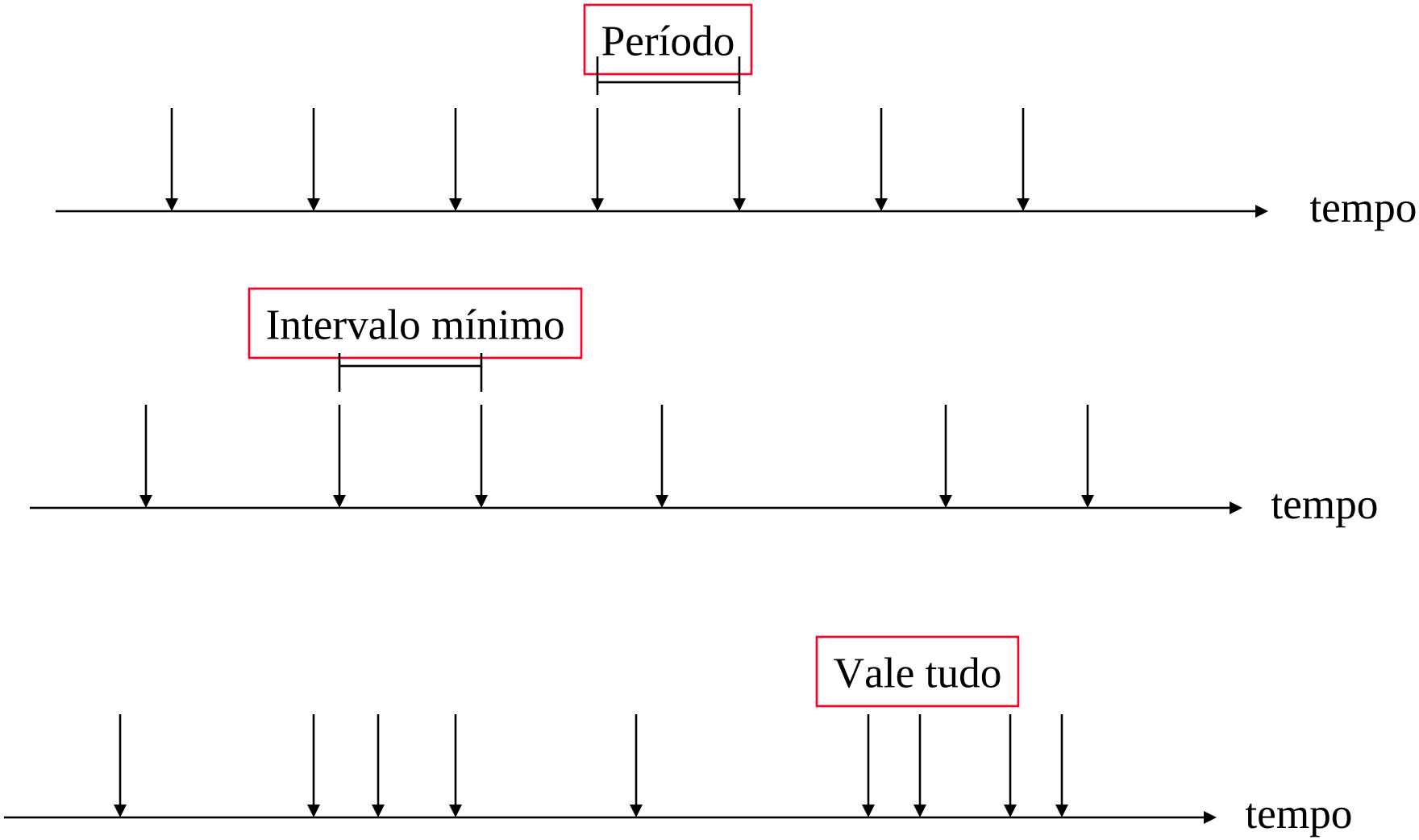
Tipos de Deadlines



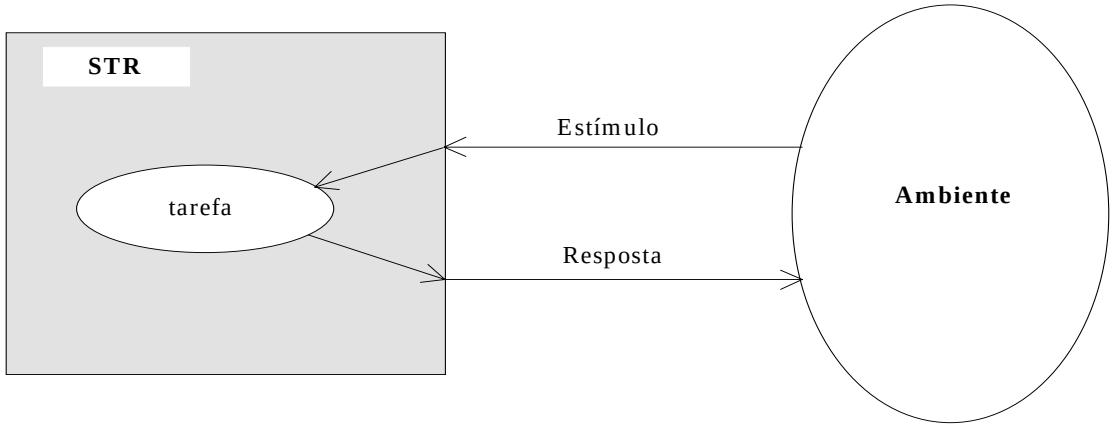
Tipos de Recorrência

- **Tarefa Periódica**
  - Tarefa é ativada a cada P unidades de tempo
  - Instantes de chegada podem ser calculados a partir do inicial
  - Exemplo: controle de processo via laço de realimentação
- **Tarefas Esporádica**
  - Instantes de chegada não são conhecidos
  - Existe um intervalo mínimo de tempo entre chegadas
  - Exemplo: atendimento a botão de alarme
- **Tarefa Aperiódica**
  - Nada é sabido quanto as ativações da tarefa
  - Exemplo: aparecimento de objeto em tela de radar

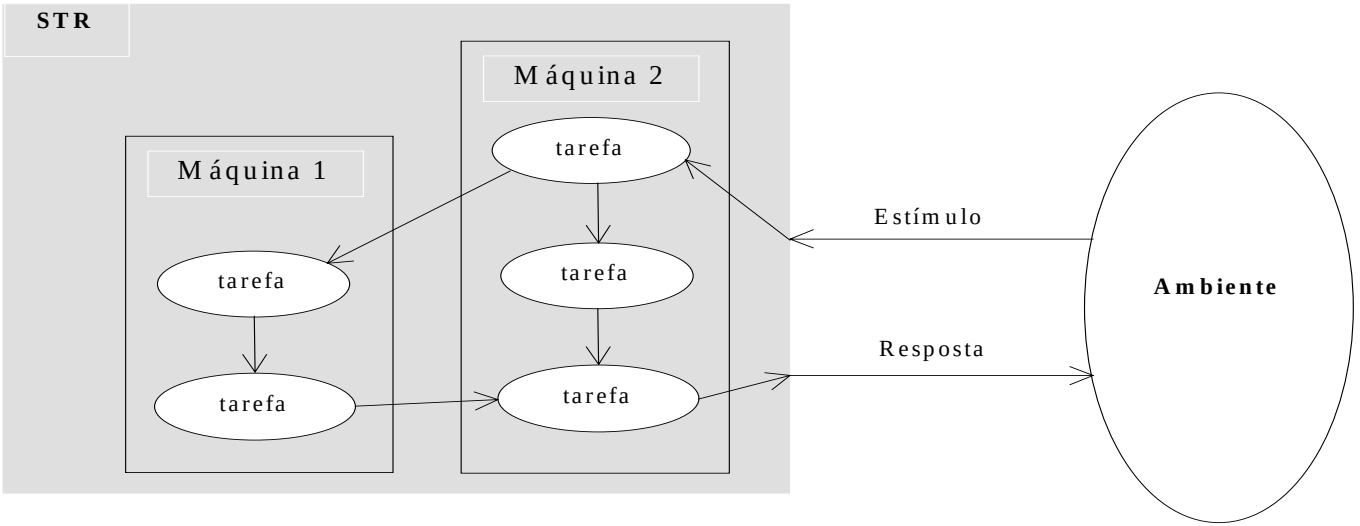
Tipos de Recorrência



Complexidade da Aplicação



Sistema Simples: Tarefa Única Responde ao Ambiente



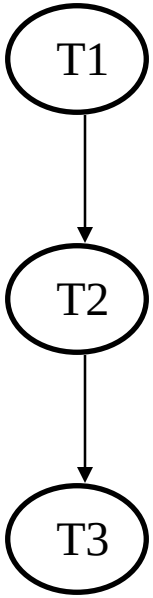
Sistema Complexo: Grafo de Tarefas Responde ao Ambiente

Relações de Precedência

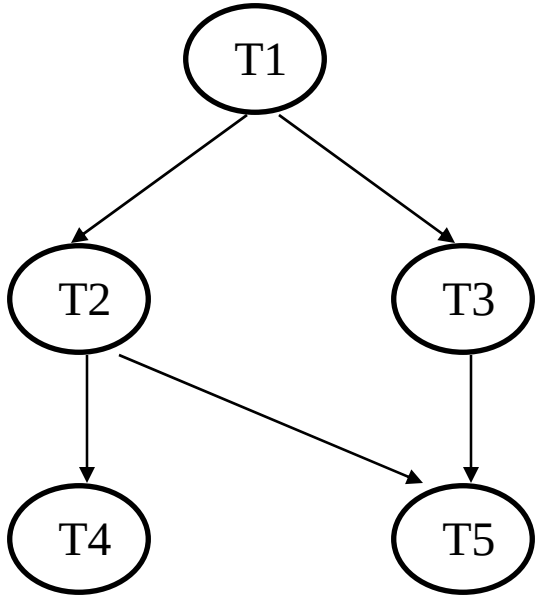
- Relações de **precedência** entre tarefas
  - Tarefa A é predecessora da tarefa B quando B somente pode iniciar depois que A estiver concluída
  - Neste caso, tarefa B é sucessora da tarefa A
  - Representado por A –> B
  - Exemplo: envio de mensagem de A para B
- **Atividade**
  - Conjunto de tarefas interligadas por relações de precedência
  - Representada por um grafo onde
    - Nodos são as tarefas
    - Flexas são as relações de precedência

Relações de Precedência

Atividade com Relações de Precedência LINEARES



Atividade com Relações de Precedência ARBITRÁRIAS



Relações de Exclusão Mútua

- Relações de **exclusão mútua** entre tarefas
  - Tarefas A e B apresentam exclusão mútua quando NÃO podem executar simultaneamente
- Exemplos:
  - Estrutura de dados compartilhada
  - Arquivo
  - Controlador de periférico

Tipos de Escalonadores

- **Escalonamento** (scheduling)
  - Identifica a forma como recursos são alocados às tarefas
- **Escalonador** (scheduler)
  - Componente do sistema responsável pela gerência dos recursos
- **Escala de Execução** (schedule)
  - Descreve quando cada tarefa ocupa cada recurso
- **Escalonamento Estático** (static scheduling)
  - Utiliza apenas informações disponíveis antes da execução
  - Capaz de oferecer previsibilidade determinista
- **Escalonamento Dinâmico** (dynamic scheduling)
  - Utiliza informações disponíveis apenas durante a execução
  - Capaz de oferecer apenas previsibilidade probabilista

Modelos de Tarefas

- **Modelo de Tarefas** (*task system*)
  - Descrição das propriedades temporais das tarefas no sistema
  - Exemplo: periódicas ou não, com duração conhecida ou não, etc
- Varia muito de sistema para sistema
- Ponto de partida para:
  - Análise de escalonabilidade
  - Escolha do suporte
    - Linguagem de programação
    - Sistema operacional
    - Arquitetura do computador
- **Carga de Tarefas** (*task load*)
  - Descrição de quais tarefas deverão ser executadas
  - Estática: Limitada e conhecida em projeto
  - Dinâmica: Conhecida somente ao longo da execução

Sistemas de Tempo Real:

Abordagens de Escalonamento

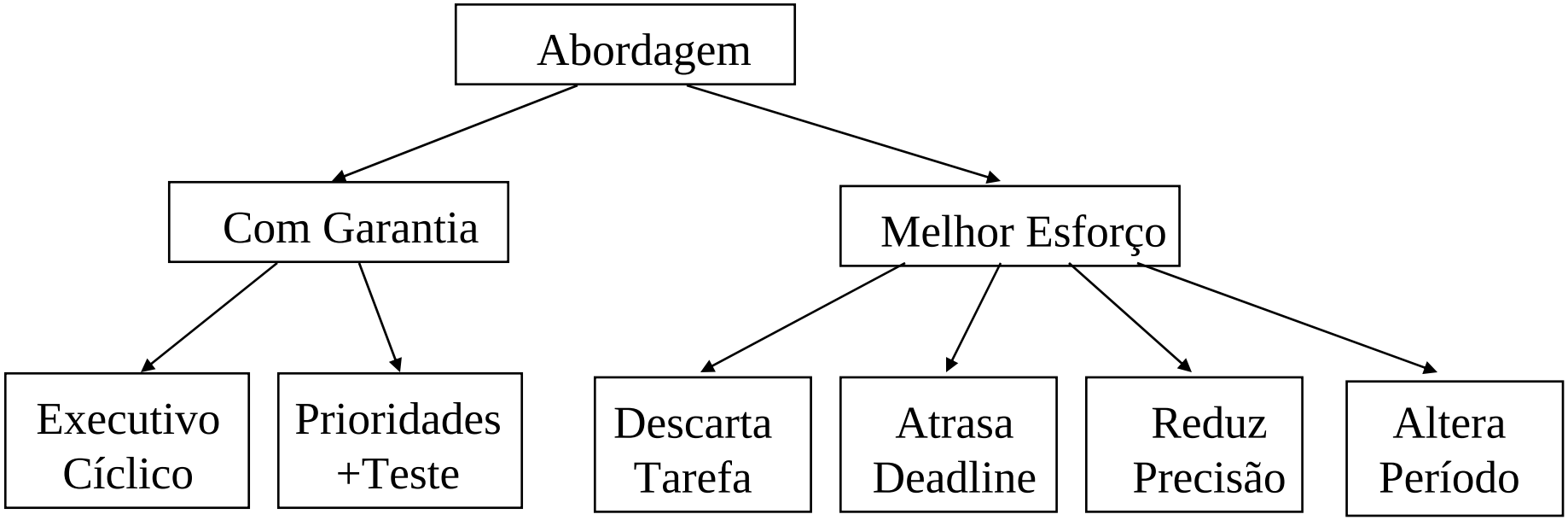
Rômulo Silva de Oliveira
Departamento de Automação e Sistemas – DAS
UFSC
romulo@das.ufsc.br
http://www.das.ufsc.br/~romulo

Necessidade de Diferentes Abordagens

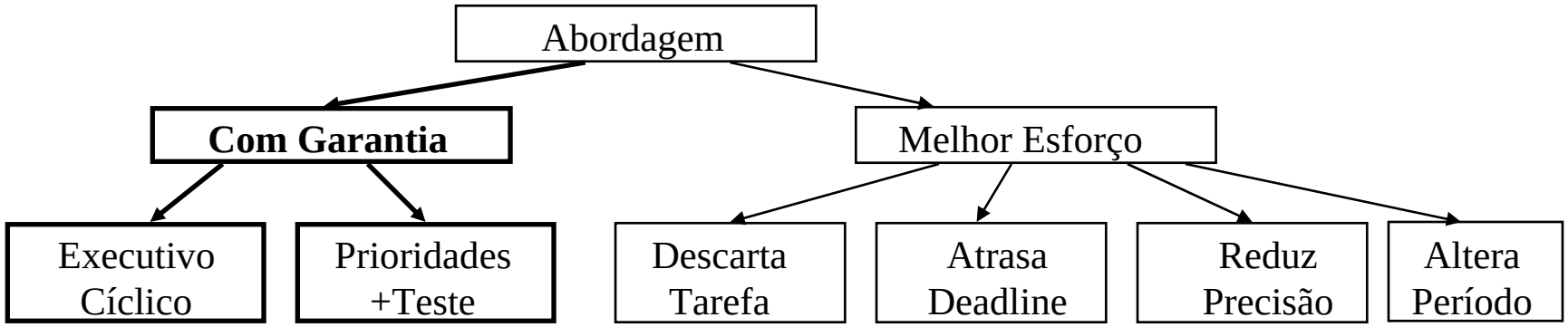
- Mercado para tempo real é amplo
- Sistemas de tempo real variam enormemente
  - Sistema de emergência em usina petroquímica
  - Controle de temperatura do freezer
  - Vídeo game
- Principais variações:
  - Crítico ou não crítico
  - Carga estática ou dinâmica
  - Importância associada com os deadlines
- Diferentes abordagens são necessárias

Classificação das Abordagens

- Abordagens básicas para o escalonamento tempo real



Abordagens com Garantia em Projeto



- Oferece previsibilidade determinista
- Análise feita em projeto
  - Carga limitada e conhecida em projeto (Hipótese de Carga)
  - Suposto um limite para faltas (Hipótese de Faltas)
- Dividida em duas partes:
  - Análise da escalonabilidade
  - Construção da escala de execução

Abordagens com Garantia em Projeto

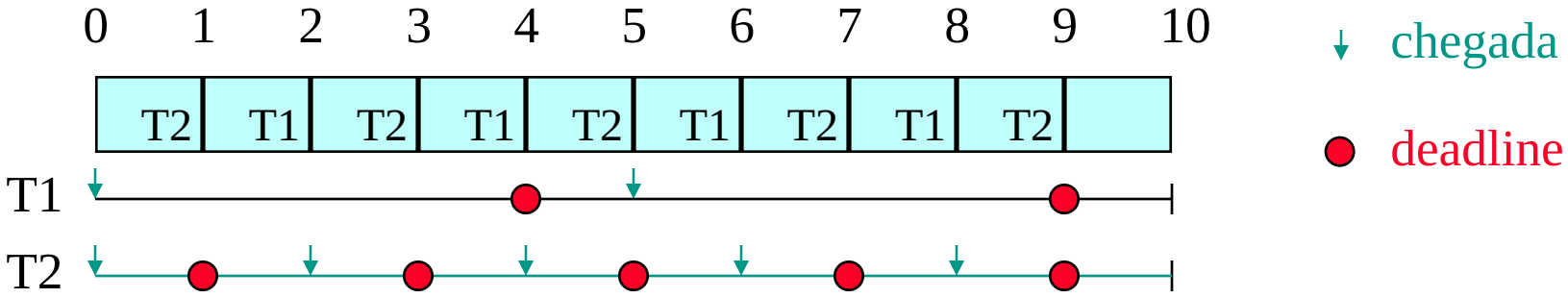
- **Vantagens**
  - Determina em projeto que todos os deadlines serão cumpridos
  - Necessário para aplicações críticas
  - Teoria serve de base para abordagens sem garantia
- **Desvantagens**
  - Necessário conhecer exatamente a carga
  - Necessário reservar recursos para o pior caso
  - Difícil determinar o pior caso em soluções COTS (commercial off-the-shelf)
  - Gera enorme sub-utilização de recursos

Executivo Cíclico

- Todo o trabalho de escalonamento é feito em projeto
- Resultado é uma **grade de execução** (time grid)
- Grade determina qual tarefa executa quando
- Garantia obtida através de uma simples inspeção da escala
- Durante a execução:
  - Pequeno programa lê a grade e dispara a tarefa aproprida
  - Quando a grade termina ela é novamente repetida
- Vantagem: Comportamento completamente conhecido
- Desvantagem: Escalonamento muito rígido, tamanho da grade
- Muito usado em aplicações embutidas (Embedded Systems)

Executivo Cíclico

- Restrições devem ser observadas na construção da grade
  - Período, tempo máximo de computação
  - Precedências, exclusões, etc
- Escalonamento pode ser:
  - Preemptivo - Tarefa pode ser suspensa e depois retomada
  - Não Preemptivo - Depois que inicia tarefa vai até o fim
- Exemplo:
  - T1: P1=5 C1=2 D1=4
  - T2: P2=2 C2=1 D2=1



Prioridades + Teste de Escalonabilidade

- Cada tarefa recebe uma prioridade
- Prioridades podem ser fixas ou variáveis
- Escalonamento em geral é preemptivo
- Teste realizado antes da execução determina escalonabilidade
  - Teste considera forma como prioridades são atribuídas
  - Complexidade depende do modelo de tarefas
- Na execução:
  - Escalonador dispara as tarefas conforme as prioridades
- Vantagem: Suporta tarefas esporádicas, não tem grade
- Desvantagem: Testes limitados a alguns modelos de tarefas
- Usado em aplicações que exigem garantia mas são muito complexas para construir uma grade

Prioridades + Teste de Escalonabilidade

- Políticas mais utilizadas:
  - **Earliest Deadline First – EDF** (prioridade variável)
  - **Rate Monotonic – RM** (prioridade fixa)
  - **Deadline Monotonic – DM** (prioridade fixa)

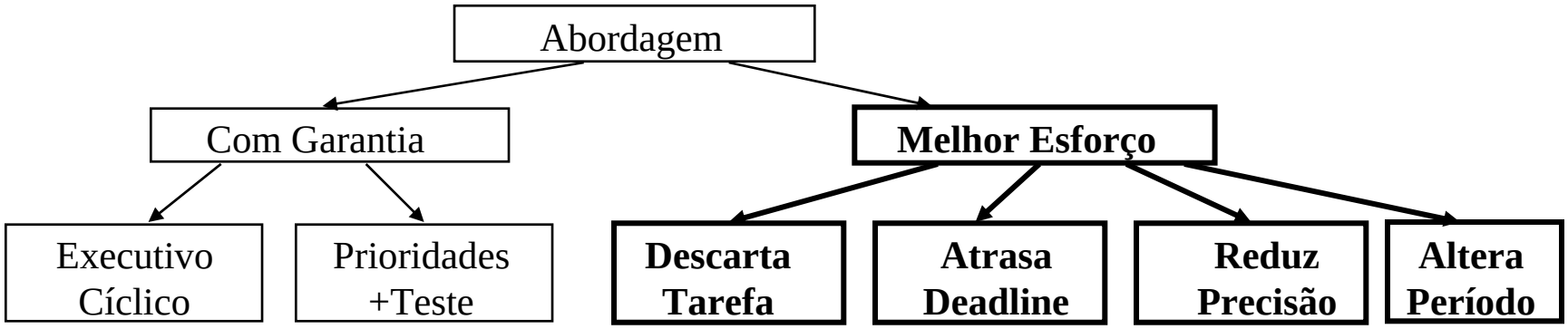
Prioridades + Teste de Escalonabilidade

- Exemplo usando RM
  - Utilização de uma tarefa:
    - Tempo máximo de computação dividido pelo período
    - T1 tem C1=12 e P1=50, então U1 = 12 / 50 = 0.24
  - Teste para Rate Monotonic, sistema é escalonável se:

$$\sum_{i=1}^{N} \left(\frac{C_i}{P_i}\right) < N\left(2^{1/N} - 1\right)$$

- Para N grandes utilização máxima tende para 69.3%
- Teste é suficiente mas não necessário

Abordagens com Melhor Esforço



- Não existe garantia que os deadlines serão cumpridos
- O “Melhor Esforço” será feito neste sentido
- Capaz de fornecer análise probabilista
  - Simulação, teoria das filas de tempo real, etc
- Algumas abordagens oferecem Garantia Dinâmica
  - Garante o deadline (ou não) no início da ativação

Abordagens com Melhor Esforço

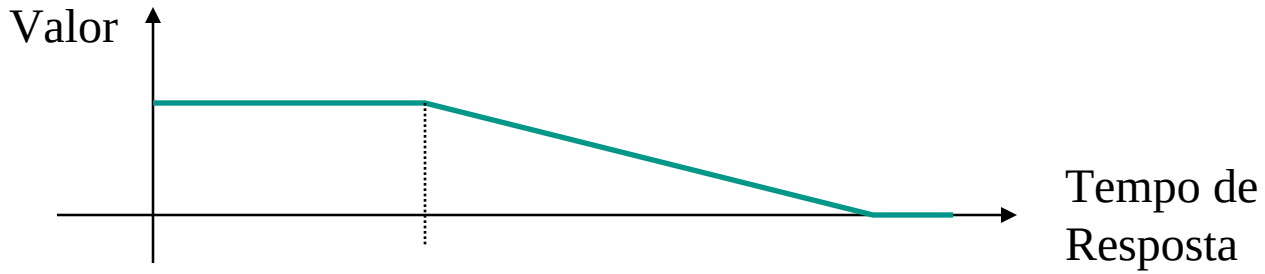
- Existe a possibilidade de Sobrecarga (*overload*)
- Sobrecarga:
  - Não é possível cumprir todos os deadlines
  - Situação natural uma vez que não existe garantia *off-line*
- Questão fundamental: Como tratar a sobrecarga ?
- Vantagens
  - Não é necessário conhecer o pior caso
  - Sistemas mais baratos, projetados para o caso médio
  - Não é necessário conhecer a carga exatamente
- Desvantagens
  - A princípio qualquer deadline poderá ser perdido

Descarte de Tarefas na Sobrecarga

- Em sobrecarga NÃO EXECUTA algumas tarefas
- As tarefas executadas cumprem o deadline
  - Mais apropriado para tarefas com deadline firm
- Pode cancelar:
  - Ativações individuais
  - Tarefas completas
- Objetivo é maximizar o número de tarefas executadas
- Tarefas podem ter importância (peso) diferentes
  - Maximiza o somatório dos pesos das tarefas executadas
- Algumas soluções utilizam um parâmetro de descarte **s**
  - Distância temporal entre ativações descartadas deve ser **s**
- Outras permitem o descarte de até **m** a cada **k** ativações

Perda de Deadlines na Sobrecarga

- Em sobrecarga ATRASA algumas tarefas
- Baseado em função tempo-valor (time-value function)
- Cada tarefa possui uma função que
  - Indica o valor da tarefa para o sistema em função do seu instante de conclusão
- Objetivo é maximizar o valor total do sistema
  - Valor do sistema é o somatório do valor das tarefas executadas
  - Tarefas podem possuir importância (peso) diferentes



Redução da Precisão na Sobrecarga

- Em sobrecarga DIMINUI a precisão de algumas tarefas
- Objetivo é “fazer o trabalho possível dentro do tempo disponível”
- Exemplos:
  - Ignorar bits menos significativos de cada pixel
  - Trabalhar com amostras de áudio menos precisas
  - Alterar resolução e tamanho de imagem na tela
  - Simplificar animações
  - Usar algoritmos de controle mais simples
  - Interromper pesquisa em algoritmos de inteligência artificial
- Aparece associada com a técnica de

  Computação Imprecisa (Imprecise Computation)

Computação Imprecisa

- Cada tarefa dividida em
  - Parte obrigatória (mandatory part)
  - Parte opcional (optional part)
- Parte Obrigatória gera resultado com qualidade mínima
- Parte Opcional refina resultado até qualidade máxima
- Podem existir tarefas com
  - apenas parte obrigatória ou
  - apenas parte opcional
- Situações normais: executa as duas partes de cada tarefa
- Sobrecarga: executa apenas a parte obrigatória
- Objetivo é maximar a qualidade total da aplicação

Alteração do Período na Sobrecarga

- Em sobrecarga aumenta o período de algumas tarefas
- Objetivo é não perder deadlines através da redução da utilização do processador que cada tarefa representa
- Exemplos:
  - Amostragem de variáveis em laço de controle
  - Taxa de apresentação dos quadros de um vídeo
  - Freqüência da atualização da tela do operador
- Chamado às vezes de

  Rate Modulation

Resumo

- Existe a necessidade de **diferentes abordagens** para o escalonamento tempo real
- Principal classificação é com respeito a **garantia**
- Abordagens com Garantia em Projeto
  - Executivo Cíclico
  - Prioridades + Teste de Escalonabilidade
- Abordagens com Melhor Esforço
  - Descarte de Tarefas
  - Perda de Deadlines
  - Redução da Precisão
  - Alteração do Período