Assignment Project Exam Help

Software System Design and Implementation

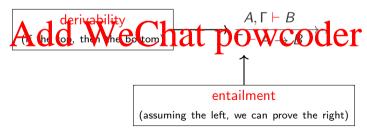
https://powcoder.com

Add Write Edit 17 14 FCS TO WW Coder

Natural Deduction

We so so that describe how to prove various connectives.

Each connective typically has introduction and elimination rules. For example, to protein simplication AW & Old Memorial B holds assuming A. This introduction rule is written as:



More rules

Implication also has an elimination rule, that is also called *modus ponens*:

Assignment Project Exam Help

Conjunction (ald) the print of prior whether deriving in the conjunction (ald) the print of prior whether deriving in the prior whether th

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \vdash B} \land \neg I$$

 $\underset{\text{It has two elimination rules:}}{Add} \overset{\Gamma\vdash A \qquad \Gamma\vdash B}{\text{WeChat powcoder}} \land \text{I}$

$$\frac{\Gamma \vdash A \land B}{\Gamma \vdash A} \land -\text{E}_1 \qquad \frac{\Gamma \vdash A \land B}{\Gamma \vdash B} \land -\text{E}_2$$

More rules

Disjunction (or) has two introduction rules:

Assignment Project Exam Help

Disjunction elimination is a little unusual:

The true literal, written \top , has only an introduction:

Add WeChat powcoder

And false, written \perp , has just elimination (ex falso quodlibet):

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash P}$$

Example Proofs

Example

ProvAssignment Project Exam Help

- \bullet $A \lor \bot \to A$

What would ne htto psqu/val prowcoder.com Typically we just define

$$eg A \equiv (A
ightarrow oldsymbol{\perp})$$

Add WeChat powcoder

Example

Prove:

- \bullet $A \rightarrow (\neg \neg A)$
- $(\neg \neg A) \rightarrow A$ We get stuck here!

Constructive Logic

The Aissigenmenter Perojecthe Exame clade due:

 $P \vee \neg P$

https://powcoder.com

Add WeChat powcoder

This is because it is a *constructive* logic that does not allow us to do proof by contradiction.

Boiling Haskell Down

The theoretical properties we will describe also apply to Haskell, but we reed a smaller language for describe purposes. TO ICCL EXAM HELD

- No user-defined types, just a small set of built-in types.
- No polymorphism (type variables)
- Just lambattps: defiponwcoderriaceom

This language is a very minimal functional language, called the simply typed lambda calculus, originally due to Alonzo Church.

Our small set of built-in types are intended to be enough to express most of the data types we would otherwise define.

We are going to use logical inference rules to specify how expressions are given types (typing rules).

Function Types

we Assignment, Project Exams: Help

https://powcoder.com

The typing rule for function application is as follows:

Add We Chat powcoder

What other types would be needed?

Composite Data Types

In addis Sife the control of the con

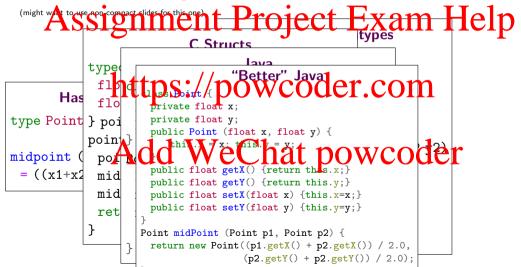
https://powcoder.com

Add WeChat powcoder

Records

Combining values conjunctively

We want to store two things in one value.



Product Types

For simply typed lambda calculus, we will accomplish this with tuples, also called product Exam Help

https://powceder.com

We won't have the declarations a med field or anything like that. More than two values can be combined by nesting products, for example a three dimensional vector:

(Int,(Int,Int))

Constructors and Eliminators

we Assignment Project Exam Help

https://powcoder.com

The only way to extract each component of the product is to use the fst and snd eliminators:

Add WeChat poxycoder

Unit Types

Assignment Project Exam Help

Currently, we have no way to express a type with just one value. This may seem useless at first, but it becomes useful in combination with other types. We'll introduce the trip spe/fyop the with other types.

We'll introduce the trip spe/fyop the with other types.

Add WeChatopowcoder

Disjunctive Composition

We can't, with the types we have, express a type with exactly three values.

data Traffic Light = Red | Amber | Green | Exam Help

In general we want to express, data that can be one of multiple alternatives, that contain differen https://powcoder.com

Example (More elaborate alternatives)

```
type Length = Int
type Angle = Atdd WeChat powcoder
         | Circle Length | Point
          Triangle Angle Length Length
```

This is awkward in many languages. In Java we'd have to use inheritance. In C we'd have to use unions.

Sum Types

We'll build in the Haskell Either type to express the possibility that data may be one of twassignment $Project\ Exam\ Help$

https://powegoder.com

These types are Also talled we chat powcoder

Our TrafficLight type can be expressed (grotesquely) as a sum of units:

TrafficLight \(\sigma \) Either () (Either () ())

Constructors and Eliminators for Sums





We can branch based on which alternative is used using pattern matching:



Examples

Assignment Project Exam Help

Our traffic light type has three values as required:

https://poweoder.eom



The Empty Type

We add another than the land of the land o

We do have a way to eliminate it, however:

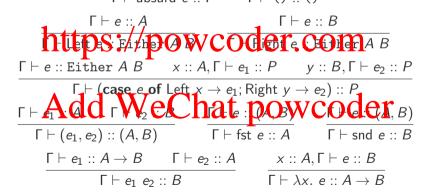
https://powcoder.com

 $\Gamma \vdash \text{absurd } e :: P$

If I have a variable of the environment of the envi

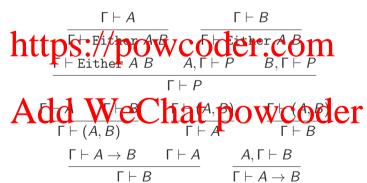
Gathering Rules

Assignmente Project Exam Help



Removing Terms...

Assignment Project Exam Help



This looks exactly like constructive logic!

If we can construct a program of a certain type, we have also created a proof of a

The Curry-Howard Correspondence

This correspondence goes by many names, but is usually attributed to Haskell Curry and William Howard ment Project Exam Help

	Programming		Logic			
1 44	Туреs		Propositions			
nttr	S Programs)V	VCO@	off.	CO	m
r	Evaluation		VCOPOT.CO Proof Simplification			

It turns out, no matter what logic you want to define, there is always a corresponding λ -calculus, and Λ cevers.

Typed λ-Calculus
Continuations
Monads
Linear Types, Session Types
Region Types

Constructive Logic
Classical Logic
Modal Logic
Linear Logic
Separation Logic

Examples

Exam Help

and Comm :: $(A, B) \rightarrow (B, A)$ and Comm p = (snd p, fst p)This proves A **https:**//powcoder.com

Example (Transitivity of Implication)

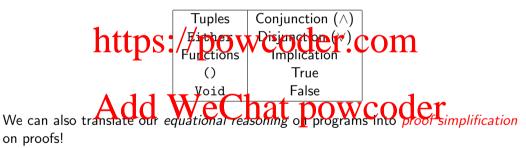
AdditiWeChat powcoder

transitive f g x = g (f x)

Transitivity of implication is just function composition.

Translating

Assignment Project Exam Help We can translate logical connectives to types and back:



on proofs!

Proof Simplification

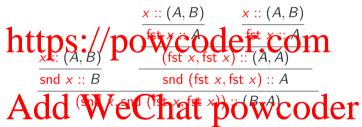
Assume Ssignment Project Exam Help
We have this unpleasant proof:

https://pow.coder.com

Add WeChat powcoder

Proof Simplification

Assum SSIGN Meetit Constraint Exam Help



We know that

$$(\operatorname{snd} x, \operatorname{snd} (\operatorname{fst} x, \operatorname{fst} x)) = (\operatorname{snd} x, \operatorname{fst} x)$$

Lets apply this simplification to our proof!

Proof Simplification

AssuAnssignment Project Exam Help

https://poxxcoder.com

Back to logic:

Add Wethat bowcoder

Applications

As ments albert, in content to the distinction between value-level and type-level languages is removed, allowing us to refer to our program in types (i.e. propositions) and then construct programs of those types (i.e. proofs) types (i.e. proofs)

Peano Arithmetic

If there's time, Liam will demo how to prove some basic facts of natural numbers in Agda, a dependently pred applies that powcoder

Generally, dependent types allow us to use rich types not just for programming, but also for verification via the Curry-Howard correspondence.

Caveats

All functions we define have to be total and terminating.

Otherwise weiget an inconsistent of that jets us profestalse things: Help

 $proof_1 :: P = NP$ $proof_1 = proof_1$

https://powcoder.com

 $proof_2 :: P \neq NF$ $proof_2 = proof_2$

Most common calculi correspond to constructive logic, not classical ones, so principles like the law of excluded middle or double negation elimination do not hold:

$$\neg \neg P \rightarrow P$$

Semiring Structure

These types we have defined form an algebraic structure called a *commutative*

SemirAge Ssignment Project Exam Help

- Associativity: Either (Either A B) $C \simeq$ Either A (Either B C)
- Identity: Either Void $A \simeq A$ Commutate The Ser A DOWNER COM

Laws for tuples and 1

- Associativity: $((A, B), C) \simeq (A, (B, C))$
- Identity: (AA) AWe Chat powcoder
 Commutativity: (A,B) \(\sum_{(B,A)} \) that powcoder

Combining the two:

- Distributivity: $(A, \text{Either } B \ C) \simeq \text{Either } (A, B) \ (A, C)$
- Absorption: (Void, A) \simeq Void

What does \simeq mean here? It's more than logical equivalence.

Isomorphism

Two types A and B are isomorphic, written $A \simeq B$, if there exists a bijection between them. This means that for each value in A we can find unique value in B and vice versa.

Example (Refactoring)

We can use this reasoning to simplify type definitions. For example data Switch Int POWCOGET. COM

Off Name

Can be simplified to the isomorphic (Name, Maybe Int).

Add We Chat powcoder

Generic Programming

Representing data types generically as sums and products is the foundation for generic programming libraries such as GHC generics. This allows us to define algorithms that work on arbitrary data structures.

Type Quantifiers

Consider the type of fast: fst Assignment Project Exam Help

This can be written more verbosely as:

Or, in a more mathematical not too wcoder.com

fst :: $\forall a \ b. \ (a,b) \rightarrow a$

This kind of quantification over the Calabastis play was polymorphism for short.

(It's also called generics in some languages, but this terminology is bad)

What is the analogue of \forall in logic? (via Curry-Howard)?

Curry-Howard

The type quantifier \forall corresponds to a universal quantifier \forall , but it is not the same as the Acceptage Mutus throughout Exam Help

First-order logic quantifiers range over a set of *individuals* or values, for example the natural numbers:

https://poweoder.com

These quantifiers range over propositions (types) themselves. It is analogous to *second-order logic*, not first-order:

Add We Chat powcoder $\forall A. \ \forall B. \ (A, B) \rightarrow (B, A)$

The first-order quantifier has a type-theoretic analogue too (type indices), but this is not nearly as common as polymorphism.

Generality

If we need a function of type Int

ppt a polymerphic function of type Value as will do just line we can just instantiate the type variable to Int. But the reverse is not true. This gives rise to an ordering.

Generality
A type A is more general transformation and appear of the instantiated to give the type B.

Example (Fundamental de Wechat powcoder

Int \rightarrow Int $\supseteq \forall z. \ z \rightarrow z \quad \supseteq \quad \forall x \ y. \ x \rightarrow y \quad \supseteq \quad \forall a. \ a$

Constraining Implementations

Assignment Project Exam Help

How many possible total, terminating implementations are there of a function of the following type?

https://powcoder.com

Add We Chat powcoder

Polymorphic type signatures constrain implementations.

Parametricity

Definition

The Angrephinien the President of polynamoful cities and the depend on values of an abstracted type.

More formally, suppose I have a polymorphic function g that is polymorphic on type a. If run any arbitrary function $f: a \to a$ on all the a values in the input of g, that will give the same result is soming printing of the couput.

Example

Add WeChat-powcoder

We know that every element of the output occurs in the input. The parametricity theorem we get is, for all f:

 $foo \circ (map \ f) = (map \ f) \circ foo$

More Examples

Assignment Project Exam Help

What's the paractipes the proposed of the proposed of the paractic prop

Example (Answer)

For any f:

Add WeChat powcoder

More Examples

Assignment Project Exam Help

 $(++):: \forall a. \ [a] \rightarrow [a] \rightarrow [a]$

What's the parametric shove powcoder.com

Example (Answer)

Add-WeChat powcoder

More Examples

Assignment Project Exam Help

concat :: $\forall a$. $[[a]] \rightarrow [a]$

What's the parametric type heover powcoder.com

Example (Answer)

Add We Chat poweoder

Assignment Project Exam Help

 $\textit{filter} :: \forall \textit{a.} \ (\textit{a} \rightarrow \textit{Bool}) \ \rightarrow [\textit{a}] \rightarrow [\textit{a}]$

What's the paracting sheet powcoder.com

Example (Answer)

Add We Chat powcoder

Parametricity Theorems

Assignment Project Exam Help

Follow a similar structure. In fact it can be mechanically derived, using the *relational* parametricity frametricity fram

Upshot: We can ask lambdabot on the Haskell IRC channel for these theorems.

Add WeChat powcoder

https://people.mpi-sws.org/~dreyer/tor/papers/wadler.pdf

Wrap-up

Assignment Project Exam Help

- That's the entirety of the assessable course content for COMP3141.
- There is a quiz for this week, but no exercise (there's still Assignment 2)

 Next week's lectures consist of a extension recture video on dependent type
- Next week's lectures consist of a extension recture video on dependent type systems, and a revision lecture on Wednesday with Curtis..
- Please come up with questions to ask Curtis for the revision lecture! It will be over very quite there exists ask Curtis for the revision lecture! It will be