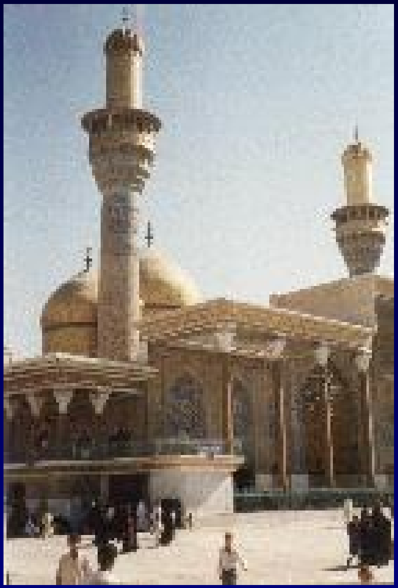
Solving the Oldest DP Problem in the World

Robert Giegerich
AG Praktische Informatik
Technische Fakultät
Bielefeld University

Excerpt from a talk given at
Mathematics of Program Construction
Dagstuhl, Germany, July 2002

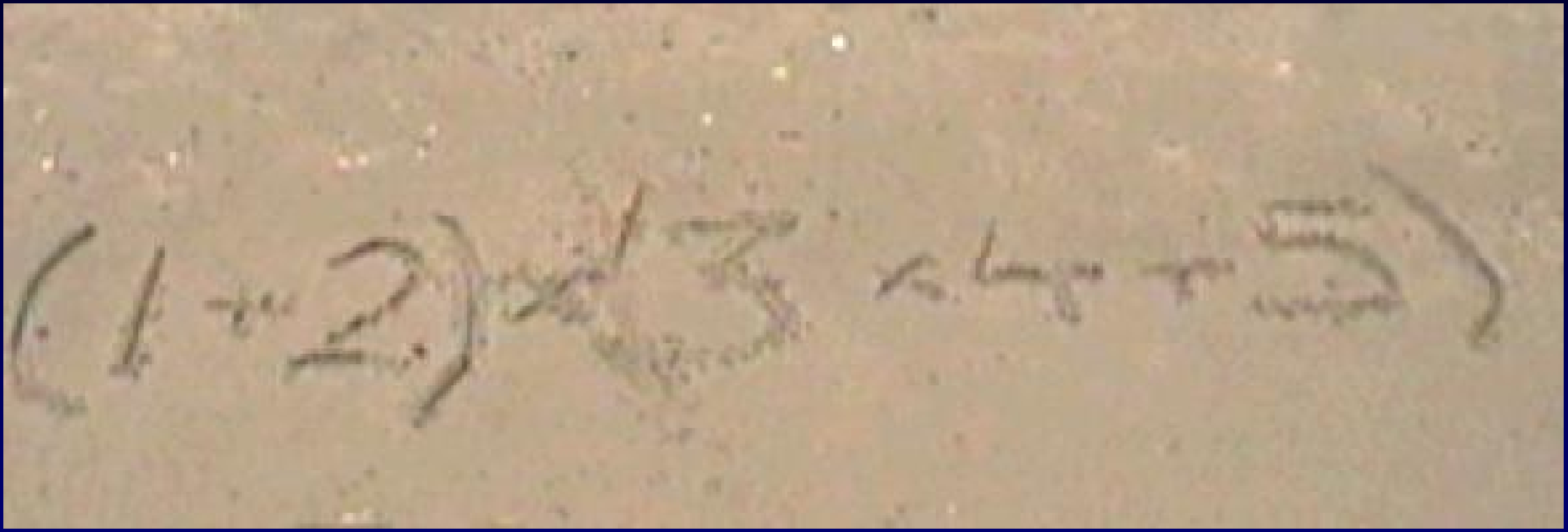
Calif El Mamun’s Caravan



Back in the old days, around the year 800, Calif El Mamun of Bagdad planned to take his two sons on their first hadj to Mekka ...

El Mamun called the camel dealer to negotiate about the required resources.

After a long day of bargaining, the bill was written in the sand.

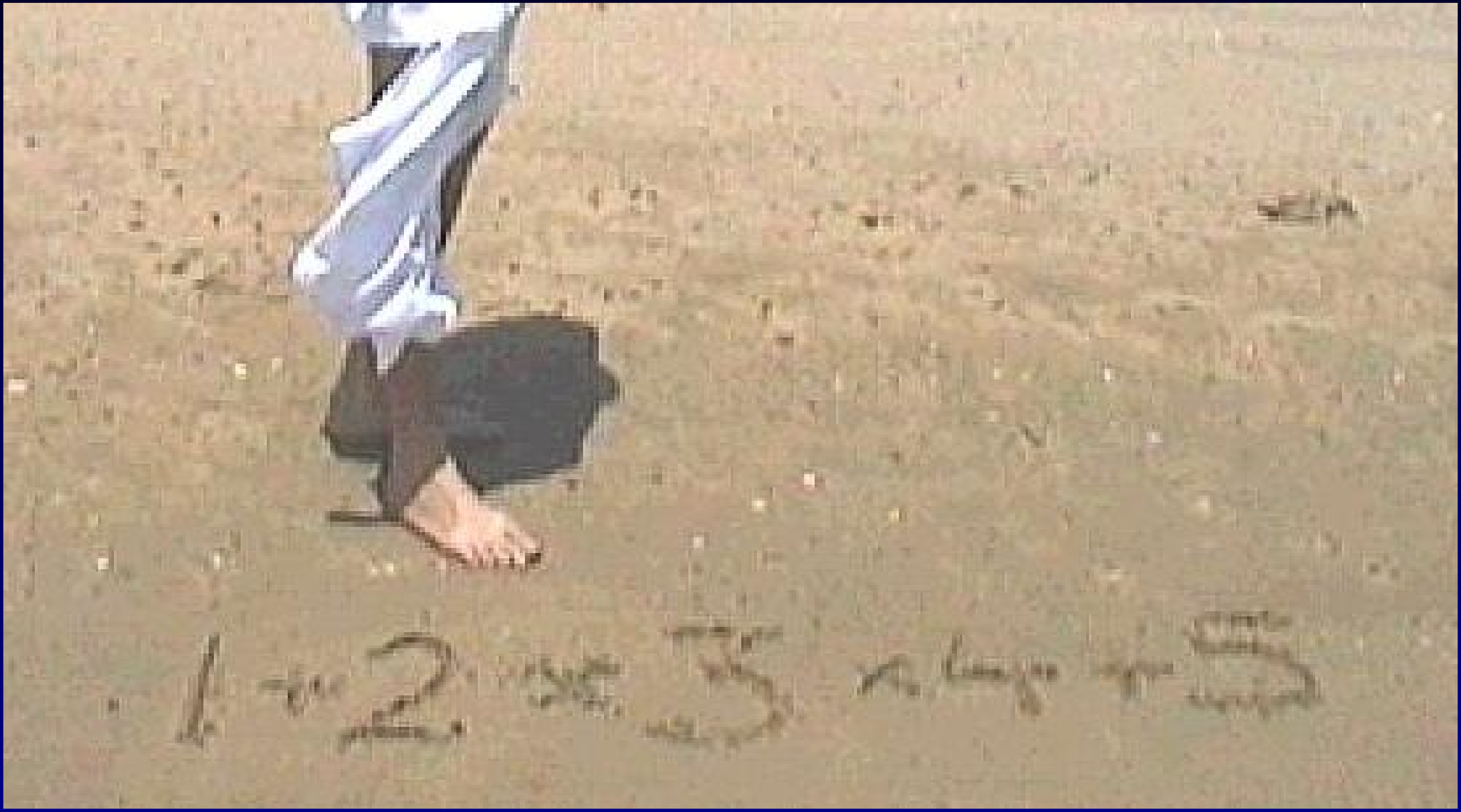


$(1+2)\times(3\times4+5)$

(1 Calif, 2 sons, 3 baggage camels for each one, costing 4 tubes of oil, plus one riding camel for each one, costing 5 tubes of oil)

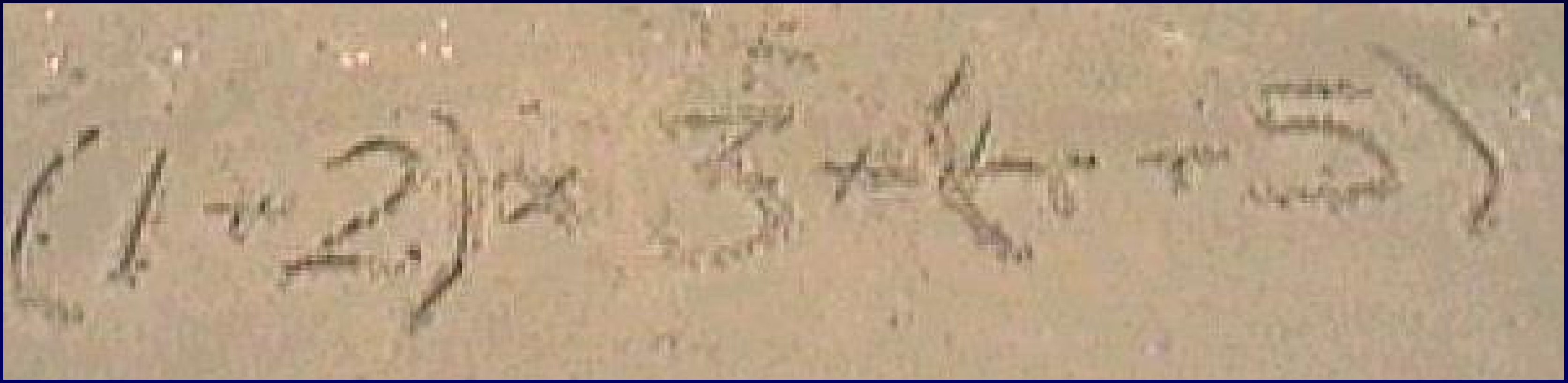
Computation was rather slow in those days ...

There came the evening prayers, and there came the wind ...



A mystery! Allah’s hand had erased the parentheses, but left untouched the rest of the formula.

The dealer was eager to redraw the parentheses:



He even claimed that now the formula showed beautiful symmetry, pleasing the eye of God.

El Mamun was not good at figures, but he knew everything about camel dealers. He felt suspicious.

El Mamun called for the mathematician. Al Chwarizmi, famous already in those days, studied the formula carefully, before he spoke.

“Some possible answers are

- $(1 + 2) * (3 * (4 + 5)) = 81$
- $1 + ((2 * (3 * 4)) + 5) = 30$
- $(1 + 2) * ((3 * 4) + 5) = 51$

which are all equal in the eyes of God.”

Apparently, Al Chwarizmi was a wise as well as a cautious man.

El Mamun’s contemplated this answer over night, and in the next morning, he ruled as follows:

- The dealer should be buried in the sand, next to his formula $(1 + 2) * 3 * (4 + 5)$.
- Someone had to take care of the dealer’s camels, and the Calif volunteered to collect them in his stables.
- Al Chwarizmi was awarded a research grant (51 tubes of oil) to study the optimal placement of parentheses, both from a buyer’s or from a seller’s perspective (depending on which side of the counter the Calif might find himself).
- Until this problem was solved, there should hold the provisional rule: “$*$ takes priority over $+$”, wherever parentheses were lacking in some formula.

Some results from this episode can still be observed today!

Some results from this episode can still be observed today!

- El Mamun became very, very rich, and his name gave rise to the word “mammon” in many modern languages.
- Studying hard, Al Chwarizmi became the father of algorithmics. He did not solve the problem, because Dynamic Programming was only developed by Richard Bellman in the 1950s.
- As a consequence of the question being unsettled, today $*$ still takes priority over $+$.

Let’s complete Al Chwarizmi’s research project using the technique of Algebraic Dynamic Programming !

The signature

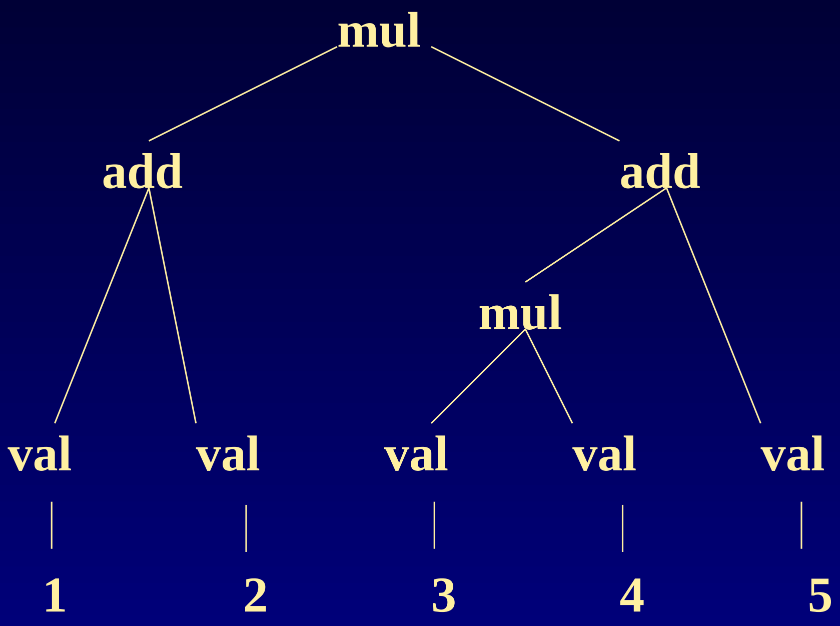
```
> module ElMamun where

> import ADPCombinators
> import Array
```

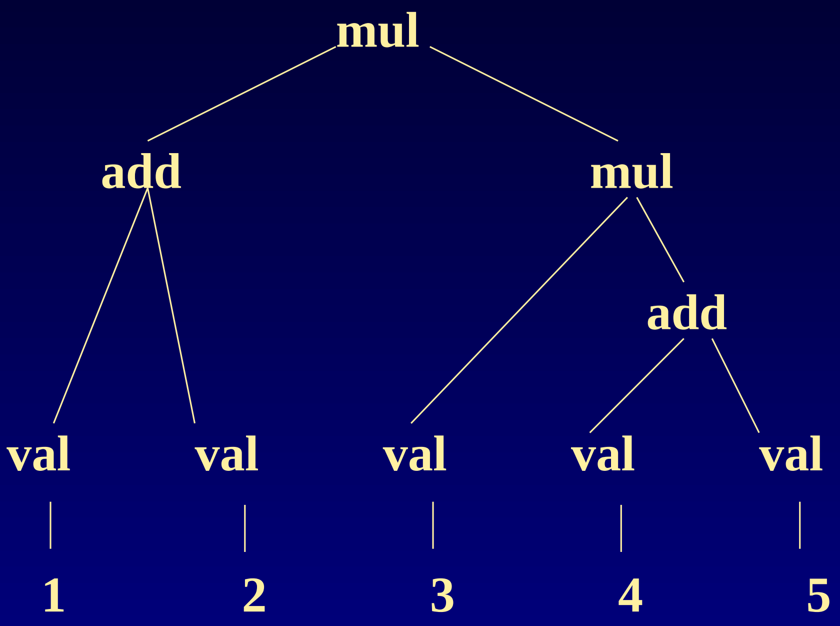
The signature:

```
> data Bill = Mult Bill Char Bill |
>             Add  Bill Char Bill |
>             Ext  Bill Char      |
>             Val  Char
```

Two candidates



the original bill



the dealer’s reconstruction

Enumeration and counting algebra

Enumeration algebra:

```
> enum = (val, ext, add, mult, h) where
>   val       = Val
>   ext       = Ext
>   add       = Add
>   mult      = Mult
>   h         = id
```

Counting algebra:

```
> count = (val, ext, add, mult, h) where
>   val c       = 1
>   ext n c     = 1
>   add  x t y  = x * y
>   mult x t y  = x * y
>   h []        = []
>   h l         = [sum l]
```

Buyer’s and seller’s algebra

The buyer’s algebra:

```
> buyer = (val, ext, add, mult, h) where
>   val c         = decode c
>   ext n c       = 10*n + decode c
>   add  x t y    = x + y
>   mult x t y    = x * y
>   h []          = []
>   h l           = [minimum l]
```

The seller’s algebra:

```
> seller = (val, ext, add, mult, h) where
>   val c        = decode c
>   ext n c      = 10*n + decode c
>   add  x c y   = x + y
>   mult x c y   = x * y
>   h []         = []
>   h l          = [maximum l]
```

The yield grammar

```
> bill alg f = axiom (p formula) where
>  (val, ext, add, mult, h) = alg

>  formula = tabulated (
>             number |||
>             add  <<< p formula ~~- plus  ~~~ p formula |||
>             mult <<< p formula ~~- times ~~~ p formula ... h)

>  number = val <<< digit ||| ext <<< number ~~- digit
>  digit  = char ’0’ ||| char ’1’ ||| char ’2’ ||| char ’3’ ||
>           char ’4’ ||| char ’5’ ||| char ’6’ ||| char ’7’ ||
>           char ’8’ ||| char ’9’

>  plus  = char ’+’
>  times = char ’*’
```

Bind input:

```
>   ff    = mk f
>   (_,n) = bounds ff

>   char         :: Char -> Parser Char
>   char c (i,j) =  [c | i+1 == j, ff!j == c]

>   tabulated = table n
>   axiom = axiom’ n
```

Some results

```
ElMamun> thebill => "1+2*3*4+5"
ElMamun> bill count thebill => [14]
ElMamun> bill enum  thebill =>
[Add (Val ’1’) ’+’ (Add (Mult (Val ’2’) ’*’ (Mult (Val ’3’)
 ’*’ (Val ’4’))) ’+’ (Val ’5’)), ... ]

ElMamun> bill buyer  thebill  => [30]
ElMamun> bill seller thebill  => [81]

ElMamun> bill time   thebill  => [12]
```

The last result is the minimal computation time (minutes) achievable by concurrent computing slaves.

Processor scheduling algebra

A good computing slave took 2 minutes for an addition, 5 for a multiplication. When the basar was booming, turnover was maximized by inserting parentheses such that computation time was minimized!

```
> time = (val, ext, add, mult, h) where
>   val c       = 0
>   ext n c     = 0
>   add  x c y  = max x y + 2
>   mult x c y  = max x y + 5
>   h []        = []
>   h l         = [minimum l]
```

For more recent developments, see the ADP home page at
http://bibiserv.techfak.uni-bielefeld.de/adp/