pset3

March 17, 2015

In [1]:
```
using JuMP, Gurobi

F = [1:4]
C = [1:12]
Γ = 48
p = 0.05

cost = [20, 10, 10, 15]

dbar = [30  35  30  35;
        40  40  40  30;
        35  40  35  40;
        30  35  35  30;
        40  45  40  30;
        30  35  35  40;
        40  25  30  30;
        30  35  35  30;
        35  25  35  30;
        35  35  50  35;
        30  35  40  40;
        35  40  45  40]
```

Out[1]:
```
12x4 Array{Int64,2}:
 30  35  30  35
 40  40  40  30
 35  40  35  40
 30  35  35  30
 40  45  40  30
 30  35  35  40
 40  25  30  30
 30  35  35  30
 35  25  35  30
 35  35  50  35
 30  35  40  40
 35  40  45  40
```

In [2]:
```
m = Model(solver=GurobiSolver(OutputFlag=0))
@defVar(m, x[F], Bin)
@defVar(m, 1 ≥ y[F,C] ≥ 0)
@defVar(m, w[F,C])
@defVar(m, q[F,C] ≥ 0)
@defVar(m, z ≥ 0)

values = Float64[]
```

Out[2]: 0-element Array{Float64,1}

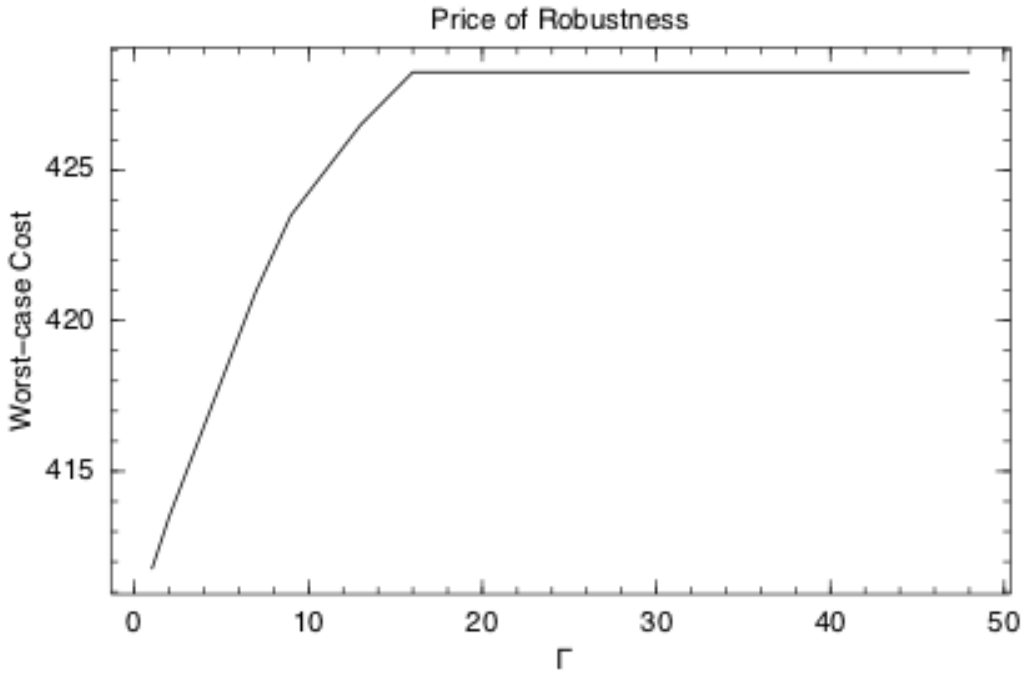
In [3]:
```
for c in C
    @addConstraint(m, sum{y[f,c], f in F} == 1)
    for f in F
        @addConstraint(m, z + q[f,c] ≥ p*dbar[c,f]*w[f,c])
        @addConstraint(m, y[f,c] ≤ x[f])
        @addConstraint(m, y[f,c] ≤ w[f,c])
        @addConstraint(m, -w[f,c] ≤ y[f,c])
    end
end
```

In [4]:
```
for γ=1:Γ
    @setObjective(m, Min, sum{cost[f]*x[f], f in F} + γ*z + sum{q[f,c]+dbar[c,f]*y[f,c], f in F
    solve(m)
    push!(values, getObjectiveValue(m))
end
```

In [5]:
```
import Winston
```

In [6]:
```
Winston.plot(values)
Winston.title("Price of Robustness")
Winston.ylabel("Worst-case Cost")
Winston.xlabel("Γ")
```

Out[6]:



In [7]:
```
function choose(n,k)
    f = 1/(2^n)
    for i=n-k+1:n # factorial(n)/(factorial(n-k)
        f *= i
    end
    for i=1:k # f/factorial(k)
        f /= i
    end
    f
end
```

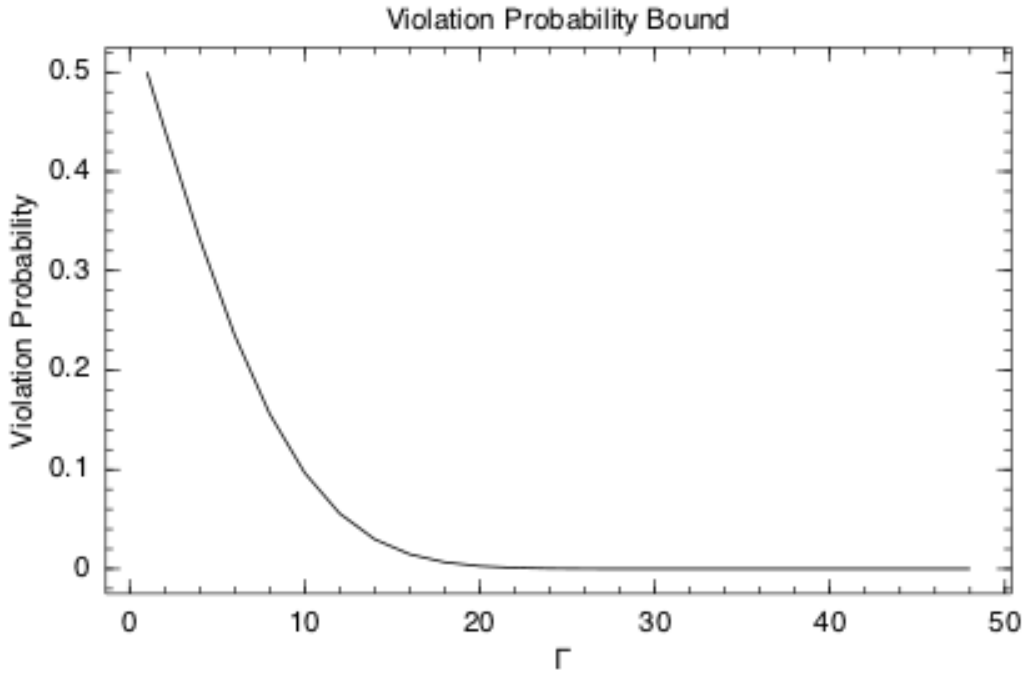
Out[7]:
```
choose (generic function with 1 method)
```

In [8]:
```
function bound(gamma)
    n = 48
    ν = (gamma+n)/2
    μ = ν-floor(ν)
    nchoosel = [choose(n, l) for l=ifloor(ν):n]
    (1-μ)*sum(nchoosel)+μ*sum(nchoosel[2:end])
end
```

Out[8]:
```
bound (generic function with 1 method)
```

In [9]:
```
Winston.plot(Float64[bound(γ) for γ=1:Γ])
Winston.title("Violation Probability Bound")
Winston.ylabel("Violation Probability")
Winston.xlabel("Γ")
```

Out[9]:



In [10]: