Final Exam Notes

Column Generation

Column generation breaks a very large problem down into a Master problem and a subproblem, where the subproblem provides "options" and the Master determines what amount of each option to use. Only options that would improve the Master problem's optimal solution will be added, keeping the problem to a manageable size.

Example: Cutting Stock

Let's say we have wires of length 107, and we need wires of lengths 3, 5, and 9. We want to meet demand while minimizing waste. Let c_i be the number of cuts of type i.

We start with the following options:

	c_1	c_2	c_3	**demand**
3	35			25
5		21		20
9			11	15
waste	2	2	8	

min. $c_1 + c_2 + c_3$ → "min waste" = "min number of source wires"

St. $35c_1 \geq 25$
$21c_2 \geq 20$
$11c_3 \geq 15$
$c_i \geq 0$

We solve this in Excel:

MASTER ITERATION 1						
vars	c1	c2	c3	LHS	sense	RHS
vals	0.71428571	0.95238095	1.36363636			
obj	1	1	1	3.03030303		
d3	35			25	>=	25
d5		21		20	>=	20
d9			11	15	>=	15

Once this is solved, we take the shadow prices and construct the subproblem. Let n_i be the number of lengths of type i per cut. Let s_i be the shadow price of the demand constraint for length i. We solve the following as an IP:

SUB ITERATION 1						
vars	n3	n5	n9	LHS	sense	RHS
vals	1	1	11			
obj	0.02857143	0.04761905	0.09090909	1.07619048		
constr	3	5	9	107	<=	107

Since the optimal LHS value is greater than 1, we know that we will get some benefit from the new cut. We then insert the cut back into the Master problem and solve again:

MASTER ITERATION 2							
vars	c1	c2	c3	c4	LHS	sense	LHS
vals	0.67532468	0.88744589	0	1.36363636			
obj	1	1	1	1	2.92640693		
d3	35			1	25	>=	25
d5		21		1	20	>=	20
d9			11	11	15	>=	15

Once again, we take the shadow prices and plug them back into the subproblem:

SUB ITERATION 2						
vars	n3	n5	n9	LHS	sense	RHS
vals	0	0	0			
obj	0.02857143	0.04761905	0.08398268	0		
constr	3	5	9	0	<=	107

Since the LHS value is less than 1, we know that there are no more cuts that will give us any benefit, so we can go back to the Master problem and solve the last formulation as an IP:

MASTER FINAL IP							
vars	c1	c2	c3	c4	LHS	sense	RHS
vals	1	1	2	0			
obj	1	1	1	1	4		
d3	35			1	35	>=	25
d5		21		1	21	>=	20
d9			11	11	22	>=	15

Dantzig-Wolfe Decomposition

Dantzig-Wolfe decomposition breaks problems down into a Master problem with multiple subproblems. The subproblems make “offers” to the Master problem, and the Master optimizes in order to determine how much it accepts each offer. Dantizig-Wolfe decomposition is used when there are constraints that can be grouped based on their variables, and is therefore classically used when multiple small LPs have been combined (eg. the merging of two power plants under one company).

The subproblem constraints define the feasible region of the subproblem that the Master problem needs to adhere to. The feasible region and corner points of the subproblem don’t change, but the Master defines a new objective for the sub with each iteration, determining which cornerpoint gets passed back to the Master. Since the Master problem only receives cornerpoints, and since all points within a feasible region are linear combinations of its cornerpoints, the Master problem can ensure that it falls within the feasible region of each subproblem by optimizing the linear combination of offers made.

Each iteration:

1. Subproblems pass corner points to the Master problem
2. Master problem optimizes based on the best combination of available cornerpoints
3. Master problem provides new objective functions to the subproblems, based on which new corner point improves the solution the most
4. Subproblems find new corner points to pass to the Master problem based on what best improves the Master value

The algorithm ends once all the subproblems can no longer improve the basis ($z = 0$).

Example: Steel Plants Merge

The Pike and Quid steel plants produce two types of steel constrained by resources.

Pike resource constraints per ton of steel product & availability.

	Iron (tons)	Coal (tons)	Furnace (tons)	Profit ($/ton)
Steel1	8	3	2	90
Steel2	6	1	1	80
Availability	N	12	10	-

Quid resource constraints per ton of steel product & availability.

	Iron (tons)	Coal (tons)	Furnace (tons)	Profit ($/ton)
Steel1	7	3	1	70
Steel2	5	2	1	60
Availability	N	15	4	-

Imagine they are purchased by the same company and have a combined iron supply of 80 tons. The problem can then be formulated as the following:

max. $90p_1 + 80p_2 + 70q_1 + 60q_2$

st. $8p_1 + 6p_2 + 7q_1 + 5q_2 \leq 80$
$3p_1 + p_2 \leq 12$
$2p_1 + p_2 \leq 10$
$3q_1 + 2q_2 \leq 15$
$q_1 + q_2 \leq 4$
$p_i, q_i \geq 0$

The constraints can clearly be grouped into sub-problems, while the objective function and iron constraint contain all the variables and therefore must remain in the Master problem.

Let's define P_i^k as the amount of steel product of type i in the k^{th} offer from the Pike plant. Similarly, let's define Q_i^k as the amount of steel product of type i in the k^{th} offer from the Quid plant. In addition, let λ^k and μ^k represent the extent to which the k^{th} offers are accepted. Thus, the Master problem can be reformulated:

max. $\Sigma \lambda^k(90P_1^k + 80P_2^k) + \Sigma \mu^k(70Q_1^k + 60Q_2^k)$
st. $\Sigma \lambda^k(8P_1^k + 6P_2^k) + \Sigma \mu^k(7Q_1^k + 5Q_2^k) + s_1 = 80$
$\Sigma \lambda^k = 1$
$\Sigma \mu^k = 1$
$\lambda^k, \mu^k, s_1 \geq 0$

In addition, the subproblems can be reformulated as the following:

Pike	Quid
max. ?????? st. $3p_1 + p_2 \leq 12$ $2p_1 + p_2 \leq 10$ $p_i \geq 0$	max. ????? st. $3q_1 + 2q_2 \leq 15$ $q_1 + q_2 \leq 4$ $q_i \geq 0$

The objective functions will be calculated as $c_{\lambda i+1}{}^T$ - (s.p.)$a_{\lambda i+1}$.

ITERATION #1

For the initial offers, we select a "null offer" where $(P_1^0, P_2^0, Q_1^0, Q_2^0) = (0, 0, 0, 0)$
We solve:

max. $\lambda^0(90(0) + 80(0)) + \mu^0(70(0) + 60(0))$
st. $\lambda^0(8(0) + 6(0)) + \mu^0(7(0) + 5(0)) + s_1 = 80$
$\lambda^0 = 1$
$\mu^0 = 1$
$\lambda^0, \mu^0, s_1 \geq 0$

By inspection, we see that the shadow prices are 0, as relaxing any constraint by 1 would have no effect. Thus, we calculate the the objective functions for the subproblems and solve as:

$c_{\lambda i+1}^T$ - (s.p.)$a_{\lambda i+1}$ $= 90P_1^1+80P_2^1$ - (0, 0, 0) $(8P_1^1 + 6P_2^1, 1, 0)^T$ $= 90P_1^1 + 80P_2^1$ max. $90P_1^1 + 80P_2^1$ st. $3P_1^1 + P_2^1 \leq 12$ $2P_1^1 + P_2^1 \leq 10$ $P_i^1 \geq 0$ $\Rightarrow P_1^1 = 0, P_2^1 = 10, z = 800$	$c_{\mu i+1}^T$ - (s.p.)$a_{\mu i+1}$ $= 70Q_1^1+60Q_2^1$ - (0, 0, 0) $(7Q_1^1 + 5Q_2^1, 0, 1)^T$ $= 70Q_1^1 + 60Q_2^1$ max. $70Q_1^1 + 60Q_2^1$ st. $3Q_1^1 + 2_2^1 \leq 15$ $Q_1^1 + Q_2^1 \leq 4$ $Q_i^1 \geq 0$ $\Rightarrow Q_1^1 = 4, Q_2^1 = 0, z = 280$

ITERATION #2

We pass the corner points back to the Master problem:

max. $\lambda^0(0) + \mu^0(0) + \lambda^1(90(0) + 80(10)) + \mu^1(70(4) + 60(0))$
st. $\lambda^0(0) + \mu^0(0) + \lambda^1(8(0) + 6(10)) + \mu^1(7(4) + 5(0)) + s_1 = 80$
$\lambda^0 + \lambda^1 = 1$
$\mu^0 + \mu^1 = 1$
$\lambda^0, \mu^0, \lambda^1, \mu^1, s_1 \geq 0$

This solves for $\lambda^0 = 0$, $\mu^0 = 2/7$, $\lambda^1 = 1$, $\mu^1 = 5/7$, $s_1 = 0$, $z = 1000$.

We get the shadow prices using commercial software so that we can calculate the the objective functions for the subproblems and solve as:

$c_{\lambda i+1}{}^T$ - (s.p.)$a_{\lambda i+1}$ $= 90P_1{}^2 + 80P_2{}^2$ - (10, 200, 0) $(8P_1{}^2 + 6P_2{}^2, 1, 0)^T$ $= 10P_1{}^2 + 20P_2{}^2 - 200$ max. $10P_1{}^2 + 20P_2{}^2 - 200$ st. $3P_1{}^2 + P_2{}^2 \leq 12$ $2P_1{}^2 + P_2{}^2 \leq 10$ $P_i{}^2 \geq 0$ $\Rightarrow P_1{}^2 = 0, P_2{}^2 = 10, z = 0$	$c_{\mu i+1}{}^T$ - (s.p.)$a_{\mu i+1}$ $= 70Q_1{}^2 + 60Q_2{}^2$ - (10, 200, 0) $(7Q_1{}^2 + 5Q_2{}^2, 0, 1)^T$ $= 0Q_1{}^2 + 10Q_2{}^2$ max. $10Q_2{}^2$ st. $3Q_1{}^2 + 2Q_2{}^2 \leq 15$ $Q_1{}^2 + Q_2{}^2 \leq 4$ $Q_i{}^2 \geq 0$ $\Rightarrow Q_1{}^2 = 0, Q_2{}^2 = 4, z = 40$

ITERATION #3

Since z = 0 for the Pike problem, there is no new offer from Pike. We pass the corner points back to the Master problem:

max. $\lambda^0(0) + \mu^0(0) + \lambda^1(800) + \mu^1(280) + \mu^2(70(0) + 60(4))$
st. $\lambda^0(0) + \mu^0(0) + \lambda^1(60) + \mu^1(28) + \mu^2(7(0) + 5(4)) + s_1 = 80$
$\lambda^0 + \lambda^1 = 1$
$\mu^0 + \mu^1 + \mu^2 = 1$
$\lambda^0, \mu^0, \lambda^1, \mu^1, \mu^2, s_1 \geq 0$

This solves for $\lambda^0 = 0$, $\mu^0 = 0$, $\lambda^1 = 1$, $\mu^1 = 0$, $\mu^2 = 1$, $s_1 = 0$, z = 1040.

We get the shadow prices using commercial software so that we can calculate the the objective functions for the subproblems and solve as:

$c_{\lambda i+1}^T$ - (s.p.)$a_{\lambda i+1}$ = $90P_1^3 + 80P_2^3$ - (12, 80, 0) $(8P_1^3 + 6P_2^3, 1, 0)^T$ = $-6P_1^3 + 8P_2^3 - 80$ max. $-6P_1^3 + 8P_2^3 - 80$ st. $3P_1^3 + P_2^3 \leq 12$ $2P_1^3 + P_2^3 \leq 10$ $P_i^3 \geq 0$ $\Rightarrow P_1^3 = 0, P_2^3 = 10, z = 0$	$c_{\mu i+1}^T$ - (s.p.)$a_{\mu i+1}$ = $70Q_1^3 + 60Q_2^3$ - (12, 80, 0) $(7Q_1^3 + 5Q_2^3, 0, 1)^T$ = $-14Q_1^3 + 0Q_2^3$ max. $-14Q_1^3$ st. $3Q_1^3 + 2Q_2^3 \leq 15$ $Q_1^3 + Q_2^3 \leq 4$ $Q_i^3 \geq 0$ $\Rightarrow Q_1^3 = 0, Q_2^3 = 0, z = 0$

There are no new offers from Pike or Quid, so the solution is the one from the previous Master problem. Thus, the amount of iron going to Pike is:

$\Sigma \lambda^k(8P_1^k + 6P_2^k)$
$= \lambda^0(8P_1^0 + 6P_2^0) + \lambda^1(8P_1^1 + 6P_2^1)$
$= 0(8(0) + 6(0)) + 1(8(0) + 6(10))$
= 60

And the amount going to Quid is:

$\Sigma \mu^k(7Q_1^k + 5Q_2^k)$
$= \mu^0(7Q_1^0 + 5Q_2^0) + \mu^1(7Q_1^1 + 5Q_2^1) + \mu^2(7Q_1^2 + 5Q_2^2)$
$= 0(7(0) + 5(0)) + 0(7(4) + 5(0)) + 1(7(0) + 5(4))$
= 20

Stochastic Programming

A stochastic program is a linear program with recourse, and is used when there is some level of uncertainty (eg. the values are not deterministic).There are two stages: a decision-making stage, where a decision is made under uncertainty; and a recourse stage, where actions are taken based on existing constraints once the uncertainty is resolved. The problems that benefit the most have uncertainty in the constraints and provide corrective action to be taken.

Example: Planting Crops

You own 500 acres and want to plant wheat (w), corn (c), and barley (b). Each has a planting cost, a purchase cost, and a selling price, and you need some minimum amount to feed your cattle.

Crop	**Planting**	**Purchase**	**Selling**	**Cattle Demand**
w	150	238	170	200
c	230	210	150	240
b	260	-	36 if ≤ 6000, 10 if > 6000	-

There are three scenarios: high, medium, or low yield. Each has an equal probability of occurring (⅓). Assume that the field yields are NOT independent.

Crop	**Low yield**	**Medium yield**	**High yield**
w	2	2.5	3
c	2.4	3	3.6
b	16	20	24

Let a_i be the acres of crop i we are planting, p_i be the tons purchased, and s_i be the tons sold. Let y_i be the yield of crop i. If we want to maximize the amount of money we make, we formulate the objective function as max. sales - planting costs - purchasing costs:

max. $170s_w + 150s_c + 36s_{eb} + 10s_{cb} - 150a_w - 230a_c - 260a_b - 238p_w - 210p_c$

st. $y_w a_w + p_w - s_w \geq 200$
$y_c a_c + p_c - s_c \geq 240$
$y_b a_b - s_{eb} - s_{cb} \geq 0$
$s_{eb} \leq 6000$
$a_w + a_c + a_b \leq 5000$
$a_i, p_i, s_i \geq 0$

Now that we have the problem formulated, we incorporate each scenario into the LP.

max. $170[\frac{1}{3} s_{w,L} + \frac{1}{3} s_{w,M} + \frac{1}{3} s_{w,H}] + 150[\frac{1}{3} s_{c,L} + \frac{1}{3} s_{c,M} + \frac{1}{3} s_{c,H}] + 36[\frac{1}{3} s_{eb,L} + \frac{1}{3} s_{eb,M} + \frac{1}{3} s_{eb,H}] + 10[\frac{1}{3} s_{cb,L} + \frac{1}{3} s_{cb,M} + \frac{1}{3} s_{cb,H}] + 150a_w - 230a_c - 260a_b - 238[\frac{1}{3} p_{w,L} + \frac{1}{3} p_{w,M} + \frac{1}{3} p_{w,H}] - 210[\frac{1}{3} p_{c,L} + \frac{1}{3} p_{c,M} + \frac{1}{3} p_{c,H}]$

st. $2a_w + p_{w,L} - s_{w,L} \geq 200$
$2.4a_c + p_{c,L} - s_{c,L} \geq 240$
$16a_b - s_{eb,L} - s_{cb,L} \geq 0$
$s_{eb,L} \leq 6000$
$2.5a_w + p_{w,M} - s_{w,M} \geq 200$
$3a_c + p_{c,M} - s_{c,M} \geq 240$
$20a_b - s_{eb,M} - s_{cb,M} \geq 0$
$s_{eb,M} \leq 6000$
$3a_w + p_{w,H} - s_{w,H} \geq 200$
$3.5a_c + p_{c,H} - s_{c,H} \geq 240$
$24a_b - s_{eb,H} - s_{cb,H} \geq 0$
$s_{eb,H} \leq 6000$
$a_w + a_c + a_b \leq 500$
$a_i, p_{ij}, s_{ij} \geq 0$

This solves for $a_w = 170$, $a_c = 80$, $a_b = 250$, and:

Low yield	Average yield	High yield
$p_{w,L} = 0$ $p_{c,L} = 48$ $s_{w,L} = 140$ $s_{c,L} = 0$ $s_{eb,L} = 4000$ $s_{cb,L} = 0$	$p_{w,M} = 0$ $p_{c,M} = 0$ $s_{w,M} = 225$ $s_{c,M} = 0$ $s_{eb,M} = 5000$ $s_{cb,M} = 0$	$p_{w,H} = 0$ $p_{c,H} = 0$ $s_{w,H} = 310$ $s_{c,H} = 48$ $s_{eb,H} = 6000$ $s_{cb,H} = 0$

Nonlinear Programming

The gradient $\nabla f(x_1, x_2, x_3)$ points in the direction of the greatest increase of the function. Essentially, it's a derivative but in R^n.

$\nabla f(x_1, x_2, x_3) = [\partial f/x_1, \partial f/x_2, \partial f/x_3]^T$

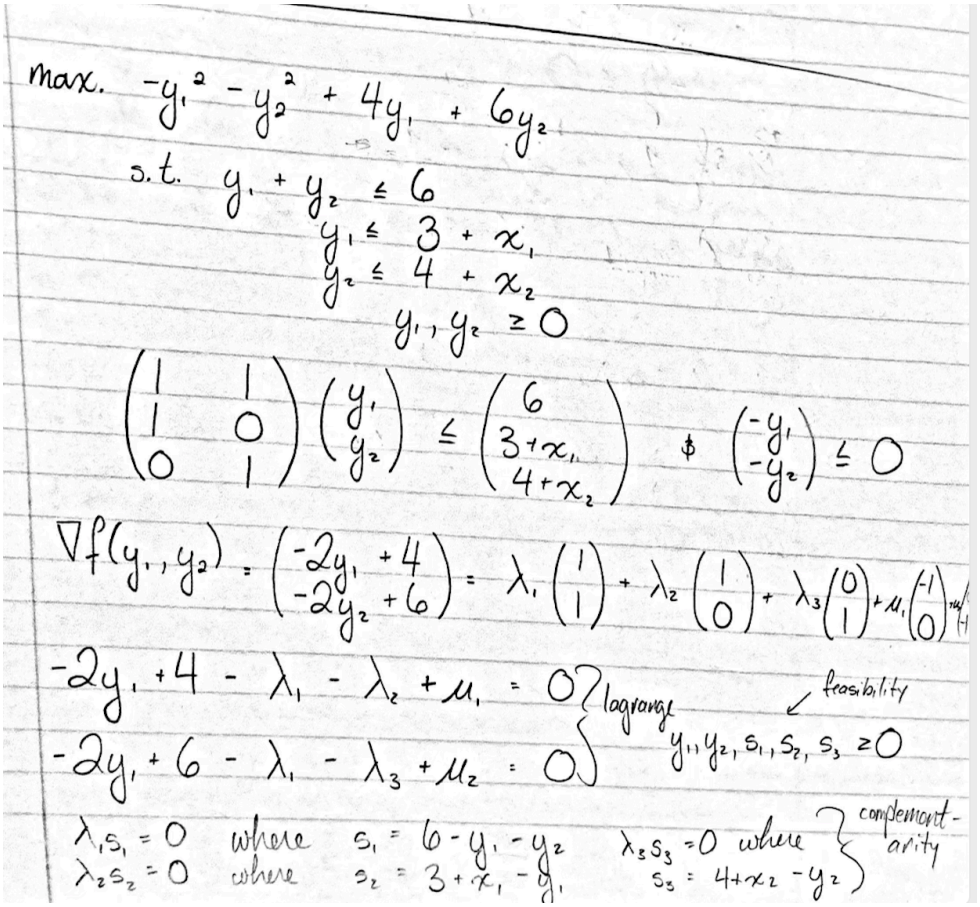
For example: $f(x, y) = x^2 + y \Rightarrow \nabla f(x, y) = [\partial f/x, \partial f/y]^T = [2x, 1]^T$

For some formulation max. f(x, y) st. g(x, y), $\nabla f(x, y) = \lambda \nabla g(x, y)$. The max point is where the gradients point in the same direction.

KKT Conditions

1. Lagrangian conditions: $\nabla f(x_1, \ldots, x_n) = \Sigma \lambda_j \nabla g_j(x_1, \ldots, x_n), \lambda_j \geq 0$
2. Complementarity conditions: $\lambda_j s_j = 0 \ \forall \ j$
3. Feasibility conditions: $s_j \geq 0 \ \forall \ j$ where $s_j = c_j - g_j(x_1, \ldots, x_n)$

Example: Simple KKT Conditions



Bi-Level Programming

Bi-Level Programming

Leader: designs the system & moves first
Follower: operates within the system to optimize their own objective & moves after the leader. The follower wants to choose a vector y assuming that the leader has first chosen some vector x.

$$\max_x \; F(x, y^*) \quad \text{s.t.} \quad x \in X$$

where $y^* = \underset{y}{\text{Arg}} \left[\max f(x, y) \text{ s.t. } y \in Y(x) \right]$

Knowing how the follower behaves will allow the leader to choose x in the most optimal manner. However, the two max functions cause problems when solving — we use the KKT conditions to replace the follower's problem.

eg. $\max_x \; -x + 4y^*$

$$\text{s.t.} \quad \begin{aligned} x + y^* &\geq 3 \\ 2x - y^* &\geq 0 \\ x &\geq 0 \end{aligned}$$

where $y^* = \underset{y}{\text{Arg}} \left[\max. \; y \quad \text{s.t.} \quad \begin{aligned} 2x + y &\leq 12 \\ 3x - 2y &\geq 4 \\ y &\geq 0 \end{aligned} \right]$

To get the KKT conditions for the follower problem, we move all constants to one side & ensure we have only $\leq$ constraints.

$$\max_{y} \quad y$$
$$\text{s.t.} \quad y \leq 12 - 2x$$
$$2y \leq -4 + 3x$$
$$-y \leq 0$$

Lagrangian condition: $\nabla f(y) = \sum_{j=1}^{m} \lambda_j \, g_j(y)$

$$1 = \lambda_1 + 2\lambda_2 - \mu$$
$$\lambda_1, \lambda_2, \mu \geq 0$$

Complementarity conditions: $\lambda_1 s_1 = 0$ where $s_1 = 12 - 2x - y$
$\lambda_2 s_2 = 0$ where $s_2 = -4 + 3x - 2y$
$\mu y = 0$

Feasibility conditions: $y, s_1, s_2 \geq 0$

We add these conditions as constraints to the leader problem:

→ we replace y^* with y b/c y satisfies KKT & is ∴ assumed to be optimal

$$\max_{x,y} \quad -x + 4y$$
$$\text{s.t.} \quad x + y \geq 3$$
$$2x - y \geq 0$$
$$2x + y + s_1 = 12$$
$$-3x + 2y + s_2 = -4$$
$$\lambda_1 + 2\lambda_2 - \mu = 1$$
$$\lambda_1 s_1 = 0; \quad \lambda_2 s_2 = 0; \quad \mu y = 0$$
$$x, y, s_1, s_2, \lambda_1, \lambda_2, \mu \geq 0$$

N-Player Games

Coalition (S): a subgroup of players that may be able to achieve what individuals cannot
Claim: the amount that each player/coalition would expect if they were the only one in the game
Value attained v(S): the value that could indisputably be assigned to each coalition without preventing players outside the coalition from acquiring their full claim
Characteristic function: the set of v(S) for all possible S
Reward vector: $x = \{x_1, x_2, \ldots, x_i\}$ where x_i is the final reward for player i

Game 1: Three friends Ahmed, Betty, and Chen live 7, 16, and 30km away and share an Uber with a rate of $3/km.

Game 2: Dante, Elad, and Fatima are owed $100, $200, and $300 by some company. When the company goes bankrupt, there is only $200 to pay them.

Proportional Splitting

In proportional splitting, each player should get the same fraction of their claim.

For game 1, if each friend travels separately, they pay a total of (3*7)+(3*16)+(3*30) = $159. However, if they travel together, they only pay a total of $90. Therefore everyone pays (90/150) of their original claim.

- Ahmed: 21 (90/159) = $11.89
- Betty: 48 (90/159) = $27.17
- Chen: 90 (90/159) = $50.94

For game 2, if each person received their full claim, they would receive a total of $600. However, since there is only $200 left, each person receives (200/600) of their original claim.

- Dante: 100 (200/600) = $33
- Elad: 200 (200/600) = $67
- Fatima: 300 (200/600) = $100

Building the Characteristic Function

1. Determine the claim for each coalition
2. Find the highest value for v(S) such that
    a. v(s) ≤ the claim of S
    b. v(s) ≤ the total amount available minus the claim of other players not in S (if negative, v(S) = 0)

For game 1, we start with the following:

S	**A**	**B**	**C**	**AB**	**AC**	**BC**	**ABC**
Claim	21	48	90	48	90	90	90
v(S)							

We know that v(S) = min(claim, total bill - claim of S'), so we can fill in the table with the following:

S	**A**	**B**	**C**	**AB**	**AC**	**BC**	**ABC**
Claim	21	48	90	48	90	90	90
v(S)	min(21, 90-90)	min(48, 90-90)	min(90, 90-48)	min(48, 90-90)	min(90, 90-48)	min(90, 90-21)	90

This comes to:

S	**A**	**B**	**C**	**AB**	**AC**	**BC**	**ABC**
Claim	21	48	90	48	90	90	90
v(S)	0	0	42	0	42	69	90

For game 2, we start with the following:

S	**D**	**E**	**F**	**DE**	**DF**	**EF**	**DEF**
Claim	100	200	300	300	400	500	600
v(S)							

We know that v(S) = min(claim, total available - claim of S'), so we fill in the table:

S	**D**	**E**	**F**	**DE**	**DF**	**EF**	**DEF**
Claim	100	200	300	300	400	500	600
v(S)	min(100, 200-500)	min(200, 200-400)	min(300, 200-300)	min(300, 200-300)	min(400, 200-200)	min(500, 200-100)	600

This comes to:

S	**D**	**E**	**F**	**DE**	**DF**	**EF**	**DEF**
Claim	100	200	300	300	400	500	600
v(S)	-300	-200	-100	-100	0	100	600

However, since some of the values are negative, we replace them with 0s:

S	**D**	**E**	**F**	**DE**	**DF**	**EF**	**DEF**
Claim	100	200	300	300	400	500	600
v(S)	0	0	0	0	0	100	200

Shapley Values

Axiom 1: Group rationality

$V(\{1, 2, \ldots, n\}) = \Sigma x_i$

Axiom 2: Player i should not be rewarded if there is no coalition to which they add value.

if $v(\{S\} \cup \{i\}) = v(S) \ \forall \ S$ then $x_i = 0$

With these axioms, the Shapley Reward Vector is:

$$x_i = \sum_{S, i/\in S} \frac{|S|!\,(n - |S| - 1)!}{n!}[v(\{S\}U\{i\}) - v(S)]$$

This corresponds to the sum of the probability of the player joining a coalition multiplied by the added value they bring to the coalition, for all coalitions they are not already in.

If we consider a 5-player game, where player 1 has already joined a coalition and player 2 is about to join. There are n! = 5! = 120 possible combinations of the players. The probability that player 2 will arrive after 1 but before 3, 4, and 5 is (1!)(3!)/120. Thus, the |S|! In the formula corresponds to the possible combinations of the players joining S before player i and (n-|S|-1)! Is the possible combinations of elements joining S after player i.

The added value is v(12) - v(1).

Let's work out the values for game 1. We have the following characteristic function:

S	**A**	**B**	**C**	**AB**	**AC**	**BC**	**ABC**
Claim	21	48	90	48	90	90	90
v(S)	0	0	42	0	42	69	90

We do each person's Shapley values individually, starting with Ahmed:

S without A	**{}**	**{B}**	**{C}**	**{BC}**
Probability	0!2! / 3! = 1/3	1!1! / 3! = 1/6	1!1! / 3! = 1/6	2!1! / 3! = 1/3
v({S}U{A}) - v(S)	0 - 0 = 0	0 - 0 = 0	42 - 42 = 0	90 - 69 = 21

$x_a = (⅓)0 + (⅙)0 + (⅙)0 + (⅓)21 = 7$

Moving on to Betty:

S without B	**{}**	**{A}**	**{C}**	**{AC}**
Probability	0!2! / 3! = 1/3	1!1! / 3! = 1/6	1!1! / 3! = 1/6	2!1! / 3! = 1/3
v({S}U{B}) - v(S)	0 - 0 = 0	0 - 0 = 0	69 - 42 = 27	90 - 42 = 48

$x_b = (⅓)0 + (⅙)0 + (⅙)27 + (⅓)48 = 20.5$

Finally, Chen:

S without C	**{}**	**{A}**	**{B}**	**{AB}**
Probability	0!2! / 3! = 1/3	1!1! / 3! = 1/6	1!1! / 3! = 1/6	2!1! / 3! = 1/3
v({S}U{C}) - v(S)	42 - 0 = 42	42 - 0 = 42	69 - 0 = 69	90 - 0 = 90

$x_c = (⅓)42 + (⅙)42 + (⅙)69 + (⅓)90 = 62.5$

This could also be done as a large table for smaller problems.

Orderings	A	B	C
ABC	0	0	90
ACB	0	48	42
BAC	0	0	90
BCA	21	0	69
CAB	0	48	42
CBA	21	27	42
Average	7	20.5	62.5

Nucleolus of a Game – Shorthand Method

1. Award initial agreed amounts to individual players (the player's v(S))
2. Distribute remaining amount evenly until a player hits their half-way point between their min (i.e. v(S)) and max (i.e. claim).
3. Continue to distribute evenly between remaining players not yet at their half-way point until each player hits their half-way point
4. Distribute remaining amount to the players furthest from their claim until they reach the same distance from their claim as another player.
5. Continue to distribute to all players equally far from their claim until all players in the game are equally far from their claim
6. Split remaining amount evenly between all players in the game

To demonstrate, we will change game 2 so that the company has $400 to award after bankruptcy. We recalculate the characteristic function:

S	D	E	F	DE	DF	EF	DEF
Claim	100	200	300	300	400	500	600
v(S)	0	0	100	100	200	100	600

From this, we see that the halfway point for D is 50, for E is 100, and for F is 200.

For Step 1, we award everyone their v(S):

Player	**D**	**E**	**F**	**To be paid**
Reward	0	0	100	500

For Step 2, we distribute evenly until one hits their halfway point.

Player	**D**	**E**	**F**	**To be paid**
Reward	50	50	150	150

For Step 3, we distribute among those that have not yet hit their halfway point. Essentially, we are attempting to bring E and F to their halfway point.

Player	**D**	**E**	**F**	**To be paid**
Reward	50	100	200	50

For Step 4, we distribute to the players farthest from their goal until they are empirically as close to their claim as another player. Here, E and F are both 100 from their claim, while D is only 50 from their claim. Thus, we distribute to E and F.

Player	**D**	**E**	**F**	**To be paid**
Reward	50	125	225	0

If we had more money, we would continue to distribute evenly until E and F were both only 50 away from their claim, then distribute evenly among D, E, and F until we ran out of money.