Optimizer Training – Use Case Example
Territory Assignment

Anaplan Optimizer

The Anaplan Optimizer aids business planning and decision-making by solving complex problems involving millions of combinations quickly to provide a solution that is either the most feasible, or the one that is closest to your configured objective, such as most profitable, fastest, or cheapest.

The Anaplan Optimizer is configured through a dashboard, which allows you to make speculative ad- hoc changes to the input parameters for immediate results. This enables the end user to create and run optimization queries without any additional expense for IT personnel. The entire process is streamlined because the model builder and the end user can perform all their work in Anaplan without need for an analyst or programmer to write code for a new application, or for the end user to export Anaplan data to an external tool, work with that tool, and then import the results into the Anaplan model.

Optimization provides the solution for a selected variable within your Anaplan model that matches your requirement, using the values of other formula cells in your model. Optimizer makes use of linear programming to automate applying a huge number of combinations in a ‘what-if’ analysis to calculate either a set objective or the most feasible solution, for example maximum profit or minimum cost.

Optimizer is an entitled service, so you need to opt-in to access the feature. Your Account Executive will assist you in structuring and formatting your enabled models.

Some problems may not result in any feasible solution (and thus no optimal solution). This can’t be detected prior to running Optimizer but an explicit error message is displayed in such cases.

What problems can you solve with Linear Programming?

Linear programming can be used to model and analyze a wide range of business problems in planning, routing, assignment, and scheduling. It can be used to solve planning problems, such as how to maximize profit, or reduce cost by creating a formula or expression, giving values to the variables that influence the problem and returning a solution.

The expression for maximizing profit or minimizing cost is known as the Objective Function, and the goal of linear programming is to optimize this Objective Function.

Optimizer has been successfully implemented across all Lines of Business, for a variety of use cases, such as:

Matching supply and demand, Sales Territory assignment, Investment portfolio balancing, Corporate Long-Range Planning, HR Staffing, etc.

Use the Optimizer Solution

Optimizer is an opt-in, additional cost component within Anaplan. Optimizer must be enabled in your workspace. Your Account Executive will help you set up Optimizer in your enabled model. The solution will be unique to your problem and constraints, and your dashboard will reflect this. Typically, each dashboard will feature:

- Input boxes where you can enter values appropriate to your scenario.
- The button on the dashboard that runs the Optimizer.
- Variables boxes where your solution is displayed.

This allows any end user with access to the Optimizer dashboard to update the input variables and run the Optimizer as often as needed.

Limitations

Some complex problems may not result in any feasible or optimal solution. This can’t be detected prior to running Optimizer. Make sure you build your dashboard in a way that alerts the end-user of inappropriate values to minimize this possibility.

Optimizer solves only linear problems. Non-linear problems, such as a transportation scheduling problem involving rush hour, where a non-linear (logarithmic) increase of vehicles eventually causes exponential delays due to traffic jams, are not in scope.

Anaplan Optimizer supports the use of three comparators in expressions:

- greater than or equal to (<=)
- equal to (=)
- less than or equal to (>=)

All line items involved in the problem must have **Time Scale** and **Versions** set as **Not Applicable**. Optimizer doesn't support **Time** or **Versions**.

The Optimizer calculates one variable for optimality or feasibility.

Running an Optimizer process provides the standard audit features. The model history shows who made the changes and when the change occurred but does not show the before-and-after values.

There is no undo. The workaround is Restoring model to historical ID. Automation of an action that runs the Optimizer is not supported by Anaplan Connect and the Anaplan API.

Performance considerations

Be aware that while the Optimizer process is running and calculating the solution, the Anaplan model is locked so we recommend that you test in a sandbox or test environment. If a problem is so complex that the Optimizer processing time is too long, consider separating the Optimization model from the production model and contact Anaplan Support to share your experience.

For performance analysis, consider isolating inputs to your optimization problem (objective, variable, and constraints) into distinct modules so that you can easily identify them.

If the use case allows, breakdown your lists into subsets and run Optimizer sequentially.

Glossary

TERM	DEFINITION
Objective Function	The expression that guides the optimization engine while it determines which assignments best support the business goal or solution, such as maximum income or minimal expense.
Variable	The value that represents the solution to the problem (sometimes called the decision variable).
Variable Data Type (Variables must have a numeric data type)	• Integer (whole number) • Real (floating point) • Binary (zero or one)
Input Data	Values necessary for computing the solution, including any constraints.
Constraint	A limit on a value, such as its maximum, minimum, or that the value can't be negative.
Upper bound, Lower bound	The maximum or minimal value for the variable.
Linear function	When a change in value to one variable causes a directly proportional corresponding increase (or decrease) in the value of the other variable. Such a relationship displays as a straight line on a graph.
Linear program	The pursuit of a solution in the form of a real number, where the Objective Function and the constraints are linear.
Integer linear program	A linear program where variables are constrained to integral values (whole numbers).
Mixed integer linear program	A linear program where only some of the variables are constrained to integral values. Other variables can be real values (decimal numbers).
Optimality	The best solution to a problem with:
Feasibility	An alternative to optimality, this offers the possible solutions to a problem with:
Time Out	The number of seconds until an Optimizer action that is processing stops and abandons progress. This value must be set when creating an Optimizer action to prevent Optimizer running indefinitely if a problem is unsolvable.

EXERCICE

Use Cases

Use Case#1: New Territories Assignment
Use Case#2: Territories Replenishment

Data Set

Lists:

Name	Parent
Geo	Total company
Region	Geo
POD (Point of Delivery)	Region
Territory	POD

Account	All
Segment	All
Industry	All
State	All

Module

Name	Functional Area	Applies to
DAT01: Account Load	Data	Accounts
INP01: Select POD to Apply	Input	POD
INP02: Pick Constraints	Input	
INP03: Territory Segment	Input	Segment
INP04: Territory Details	Input	Territory
OUT01: Territory Details	Output	Territory

Use Case#1 New Territories Assignment

Step by Step

1/ Set-up Subset lists

Create Subset lists to run Optimizer sequentially and reduce processing time. Identify a way to breakdown Account and Territory lists. In our example, we know that an Account can only be assigned to a Territory within the same point of delivery (POD).

a)

Open module: **DAT01: Account Load**
Create line item: **ss Account: AMS**

- Format: Boolean
- Formula: 'INP01: Select POD to Apply'.Apply[lookup:POD]

b)

Open module: **DAT01: Account Load**
Create line item: **ss Account: Unassigned AMS**
Format: Boolean

- Formula: Unassigned Account AND 'ss Account: AMS'

c)

Open list: Account
Create Subsets **ss Account: AMS** and **ss Account: Unassigned AMS**
Load Subsets from **DAT01: Account Load**

d)

Open module: **INP04: Territory Details**
Create line item: **ss Territory: POD**

- Format: Boolean
- Formula: 'INP01: Select POD to Apply'.Apply[lookup:POD]

e)

Open module: **INP04: Territory Details**
Create line item: **ss Territory: AMS New**

- Format: Boolean
- Formula: **New Territory AND 'ss Territory: AMS'**

f)

Open list: Territory
Create Subsets **Territory: AMS** and **ss Territory: AMS New**
Load Subsets from **INP04: Territory Details**

We will use the Subset **ss Account: AMS** and **Territory: AMS** for existing Territories replenishment and **ss Account: Unassigned AMS** and **ss Territory: AMS New** to assign accounts to new Territories.

g)

Open Actions: create 2 processes, **Territory Assignment AMS** and **Territory Assignment AMS New**. Add the 2 Subset creation actions to the 2 processes.

2/ Create your Optimizer Calculation module (New Territories only)

a)

Create a module

Name	Functional Area	Applies to
OPT01: CAL Optimizer_AMS New	Calcul	ss Account: Unassigned AMS, ss Territory: AMS New

b)

Create Line items

Name	Format	Formula
Variable	Number	
Objective	Number	Variable*'DAT01: Account Load'.Account Potential
POD Constraint Staging	Number	IF 'INP04: Territory Details'.POD = 'DAT01: Account Load'.POD THEN 1 ELSE 0
State Constraint Staging	Number	IF ISBLANK('INP04: Territory Details'.State) OR 'INP04: Territory Details'.State = 'DAT01: Account Load'.State THEN 1 ELSE 0
Industry Constraint Staging	Number	IF ISBLANK('INP04: Territory Details'.Industry) OR 'INP04: Territory Details'.Industry = 'DAT01: Account Load'.Industry THEN 1 ELSE 0
Segment Constraint Staging	Number	IF ISBLANK('INP04: Territory Details'.Segment) OR 'INP04: Territory Details'.Segment = 'DAT01: Account Load'.Segment THEN 1 ELSE 0
Territory New	List.Territory	IF Variable > 0 THEN ITEM(Territory) ELSE BLANK

3/ Create your Optimizer Constraint module (New Territories only)

a)

Create a module

Name	Functional Area	Applies to
OPT02: SYS Constraints_AMS New	System	ss Account: Unassigned AMS, ss Territory: AMS New

b)

Create Line items

Name	Format/Summary	Formula	Applies to
POD Constraint	Boolean/All	'OPT01: CAL Optimizer_AMS New'.Variable <= 'OPT01: CAL Optimizer_AMS New'.POD Constraint Staging	ss Account: Unassigned AMS, ss Territory: AMS New
State Constraint	Boolean/All	'OPT01: CAL Optimizer_AMS New'.Variable <= 'OPT01: CAL Optimizer_AMS New'.State Constraint Staging	ss Account: Unassigned AMS, ss Territory: AMS New
Industry Constraint	Boolean/All	'OPT01: CAL Optimizer_AMS New'.Variable <= 'OPT01: CAL Optimizer_AMS New'.Industry Constraint Staging	ss Account: Unassigned AMS, ss Territory: AMS New
Segment Constraint	Boolean/All	'OPT01: CAL Optimizer_AMS New'.Variable <= 'OPT01: CAL Optimizer_AMS New'.Segment Constraint Staging	ss Account: Unassigned AMS, ss Territory: AMS New
Max Account Constraint	Boolean/All	ISBLANK('INP04: Territory Details'.Segment) OR 'INP03: Territory Segment'.Max Accounts[LOOKUP: 'INP04: Territory Details'.Segment] >= 'OPT01: CAL Optimizer_AMS New'.Variable	ss Territory: AMS New
Unique Assignment	Boolean/All	'OPT01: CAL Optimizer_AMS New'.Variable <= 1	ss Account: Unassigned AMS

3/ Create your Optimizer Option

a)

Open Actions: New Action -> Optimizer

Button Text: Territory Assignment AMS New
Problem: Linear Programming
Objective: Maximize

Objective Line Item: OPT01: CAL Optimizer_AMS New.Objective
Variable 1: OPT01: CAL Optimizer_AMS New.Variable
Variable 1 Format: Binary (0-1)
Constraint 1: OPT02: SYS Constraints_AMS New.POD Constraint
Constraint 2: OPT02: SYS Constraints_AMS New.State Constraint
Constraint 3: OPT02: SYS Constraints_AMS New.Industry Constraint
Constraint 4: OPT02: SYS Constraints_AMS New.Segment Constraint
Constraint 5: OPT02: SYS Constraints_AMS New.Max Account Constraint
Constraint 6: OPT02: SYS Constraints_AMS New.Unique Assignment

b)
Add Optimizer action to Process **Territory Assignment AMS New** and publish Action on Dashboard **New Territory Assignment** (under III. Data Input)

4/ Data Output

a)
Open module: **OUT01: New Territory Details** and add formula:

Name	Format/Summary	Formula
New Accounts#	Number/None	'OPT01: CAL Optimizer_AMS New'.Variable
New Accounts Potential	Number/None	'OPT01: CAL Optimizer_AMS New'.Objective

b)
Open Dashboard **New Territory Assignment** and Run Optimizer Action

5/ Pick Constraint to apply

Allow user input to pick Constraint to apply. It will allow more flexibility to end user, and make Optimizer testing easier.

a)
Update the following Line items in module: **OPT02: SYS Constraints_AMS New**

Name	Format/Summary	Formula	Applies to
State Constraint	Boolean/All	'INP02: Pick Constraints'.State = FALSE OR 'OPT01: CAL Optimizer_AMS New'.Variable <= 'OPT01: CAL Optimizer_AMS New'.State Constraint Staging	ss Account: Unassigned AMS, ss Territory: AMS New

Industry Constraint	Boolean/All	'INP02: Pick Constraints'.Industry = FALSE OR 'OPT01: CAL Optimizer_AMS New'.Variable <= 'OPT01: CAL Optimizer_AMS New'.Industry Constraint Staging	ss Account: Unassigned AMS, ss Territory: AMS New
Segment Constraint	Boolean/All	'INP02: Pick Constraints'.Segment = FALSE OR 'OPT01: CAL Optimizer_AMS New'.Variable <= 'OPT01: CAL Optimizer_AMS New'.Segment Constraint Staging	ss Account: Unassigned AMS, ss Territory: AMS New

b)
Open Dashboard **New Territory Assignment** and Run Optimizer Action

6/ Create your Optimizer Output module

a)
Create an Output module to host Optimizer results. The module should not apply to subsets, so you don't lose any data when you run the process for another POD.

Name	Functional Area	Applies to
OPT03: OUT Optimizer_AMS	Output	Accounts

b)
Create Line Items

Name	Format/Summary	Formula
Territory	List.Territory	
Territory New	List.Territory	

c)
Import **OPT01: CAL Optimizer_AMS New.Territory New** into **Territory New**

Use Case 2: Territories replenishment

Step by Step

1/ Copy existing modules

a)
Copy **OPT01: CAL Optimizer_AMS New** and update

Name	Functional Area	Applies to
OPT04: CAL Optimizer_AMS	Output	ss Account: AMS, ss Territory: AMS

b)
Create Line Item in **OPT04: CAL Optimizer_AMS**

Name	Format/Summary	Formula	Applies to
Current Mapping	Boolean/none	ITEM(Territory) = 'DAT01: Account Load'.Territory	ss Account: AMS, ss Territory: AMS
Current Mapping Staging	number	IF Current Mapping THEN 1 ELSE 0	ss Account: AMS, ss Territory: AMS

c)
Copy **OPT01: CAL Optimizer_AMS New** and update

Name	Functional Area	Applies to
OPT05: SYS Constraints_AMS	System	ss Account: AMS, ss Territory: AMS

d)
Update Line items

Name	Format/Summary	Formula	Applies to
POD Constraint	Boolean/All	'OPT04: CAL Optimizer_AMS'.Variable <= 'OPT04: CAL Optimizer_AMS'.POD Constraint Staging	ss Account: AMS, ss Territory: AMS
State Constraint	Boolean/All	'INP02: Pick Constraints'.State = FALSE OR 'OPT04: CAL Optimizer_AMS'.Variable <= 'OPT04: CAL Optimizer_AMS'.State Constraint Staging	ss Account: AMS, ss Territory: AMS
Industry Constraint	Boolean/All	'INP02: Pick Constraints'.Industry =	ss Account: AMS, ss Territory: AMS

		FALSE OR 'OPT04: CAL Optimizer_AMS'.Variable <= 'OPT04: CAL Optimizer_AMS'.Industry Constraint Staging	
Segment Constraint	Boolean/All	'INP02: Pick Constraints'.Segment = FALSE OR 'OPT04: CAL Optimizer_AMS'.Variable <= 'OPT04: CAL Optimizer_AMS'.Segment Constraint Staging	ss Account: AMS, ss Territory: AMS
Max Account Constraint	Boolean/All	ISBLANK('INP04: Territory Details'.Segment) OR 'INP03: Territory Segment'.Max Accounts[LOOKUP: 'INP04: Territory Details'.Segment] >= 'OPT04: CAL Optimizer_AMS'.Variable	ss Territory: AMS
Unique Assignment	Boolean/All	'OPT04: CAL Optimizer_AMS'.Variable <= 1	ss Account: AMS,

e)

Create line item in **OPT05: SYS Constraints_AMS**

Name	Format/Summary	Formula	Applies to
Current Mapping Constraint	Boolean/All	'OPT04: CAL Optimizer_AMS'.Current Mapping Staging <= 'OPT04: CAL Optimizer_AMS'.Variable	ss Account: AMS, ss Territory: AMS

2/ Create your Optimizer Option

a)

Open Actions: New Action -> Optimizer

Button Text: Territory Assignment AMS
Problem: Linear Programming
Objective: Maximize
Objective Line Item: OPT04: CAL Optimizer_AMS.Objective
Variable 1: OPT04: CAL Optimizer_AMS.Variable

Variable 1 Format: Binary (0-1)
Constraint 1: OPT05: SYS Constraints_AMS.POD Constraint Constraint 2: State Constraint
Constraint 2: OPT05: SYS Constraints_AMS.State Constraint
Constraint 3: OPT05: SYS Constraints_AMS.Industry Constraint
Constraint 4: OPT05: SYS Constraints_AMS.Segment Constraint
Constraint 5: OPT05: SYS Constraints_AMS.Max Account Constraint
Constraint 6: OPT05: SYS Constraints_AMS.Unique Assignment

b)

Add Optimizer process to **Territory Assignment** and display it to Dashboard **Territory Replenishment**.

3/ Data Output

a)

Open module: **OUT01: New Territory Details** and add formula:

Name	Format/Summary	Formula
Accounts#	Number/None	'OPT04: CAL Optimizer_AMS'.Variable
Accounts Potential	Number/None	'OPT04: CAL Optimizer_AMS'.Objective

b)

Open Dashboard **Territory Replenishment** and Run Optimizer Action

4/ Load data into Optimizer Output module

Import **OPT05: CAL Optimizer_AMS.Territory** into **OPT03: OUT Optimizer_AMS.Territory**