Oracle hard-parse vs. soft parse

Oracle SQL is parsed before execution, and checked for syntax (and parts of the semantic check) before the SQL is loaded into the library cache. As opposed to a soft parse (which does not require loading into the shared pool), a hard parse includes these steps:

Loading into shared pool - The SQL source code is loaded into RAM for parsing. (the "hard" parse step)

Syntax parse - Oracle parses the syntax to check for misspelled SQL keywords.

Semantic parse - Oracle verifies all table & column names from the dictionary and checks to see if you are authorized to see the data.

Query Transformation - Oracle will transform complex SQL into simpler, equivalent forms and replace aggregations with materialized views, as appropriate. In earlier releases of Oracle the query_rewrite=true parameter had to be set for materialized view rewriting.
Optimization - Oracle then creates an execution plan, based on your schema statistics (or maybe with statistics from dynamic sampling in 10g). Oracle build the decision tree of costs during this period, choosing the path with the lowest perceived cost.
Create executable - Oracle builds an executable file with native file calls to service the SQL query.

Fetch rows - Oracle then executes the native calls to the data files to retrieve the rows and passes them back to the calling program.

Anytime a session issues SQL statement that does not already exist in the shared pool, then Oracle has to do a hard parse. essentially performing all of the above steps. If the statement already exists, then a soft parse occurs, skipping step 1.

In a soft parse, Oracle must still perform a syntax parse and semantic check because it is possible that a DDL change altered one of the target tables or views since the SQL statement was originally executed. In the case of DDL, all related SQL is marked as invalidated.

Oracle gives us the shared_pool_size parm to cache SQL so that we don't have to parse, over-and-over again. However, SQL can age-out if the shared_pool_size is too small or if it is cluttered with non-reusable SQL (i.e. SQL that has literals "where name = "fred") in the source. Hence, the shared_pool_size parameter (memory_target in 12c AMM) has an impact of hard parses, because re-entrant SQL can age out of the shared pool.

What the difference between a hard parse and a soft parse in Oracle? Just the first step, step 1 as shown in red, above. In other words, a soft parse does not require a shared pool reload (and the associated RAM memory allocation).

A general high "parse call" (> 10/sec.) indicates that your system has many incoming unique SQL statements, or that your SQL is not reentrant (i.e. not using bind variables).

A hard parse is when your SQL must be re-loaded into the shared pool. A hard parse is worse than a soft parse because of the overhead involved in shared pool RAM allocation and memory management. Once loaded, the SQL must then be completely re-checked for syntax & semantics and an executable generated.

Excessive hard parsing can occur when your shared_pool_size is too small (and reentrant SQL is paged out), or when you have non-reusable SQL statements without host variables.

See the cursor_sharing parameter for a easy way to make SQL reentrant and remember that you should always use host variables in you SQL so that they can be reentrant.