This is a tentative proposal to shorten the PL/I calling sequence by not generating descriptors (and descriptor pointers) for all arguments to a procedure when only some of its parameters require descriptors. (The argument list would still contain room for all of the descriptor pointers.) Procedures declared options (variable) would still receive a full set of descriptors. Presently the PL/I compiler generates no descriptors if none of the parameters require them, or a full descriptor set if one or more of the parameters has star extents.

The following are positive implications of the proposed change:

- calling sequences to procedures having some but not all parameters requiring a descriptor would be shorter by two words for each parameter not requiring a descriptor.

- compilation of these calls would be faster.

The following are some negative implications of the proposed change:

- some debugging routines such as print_arg_list and list_arg, used by trace, trace_stack, and debug assume that if descriptors are provided, every argument has a descriptor. If these programs were not changed, they could fault looking for the missing descriptor pointers.
certain illegal argument-parameter mismatches that work now would fail causing faults. For example:

caller:  proc;
dcl   foo entry(char(*) , char(32) varying);
   .
   .
call foo(a,b);
end;

foo: proc(a,b);
dcl   a char(*);
dcl   b char(*) varying;
   .
   .
end;

One solution for the first negative implication would be to have a bit in the argument list header indicate whether an incomplete descriptor list is provided.

Obviously before anything is done to implement this proposal we need more discussion. I would appreciate comments including other positive and negative implications and solutions for negative implications as well as opinions on whether this is even worth doing.